

Analyzing NYPD Stop, Question, and Frisk  
with Machine Learning Techniques



A Thesis Submitted in Partial Fulfillment of the Requirements  
for the Degree of Master of Engineering in Industrial Engineering  
Department of Industrial Engineering  
FACULTY OF ENGINEERING  
Chulalongkorn University  
Academic Year 2021  
Copyright of Chulalongkorn University

การวิเคราะห์ปฏิบัติการเรียกสกัดจับสอบถามและค้นตัวของกรมตำรวจนิวยอร์ก  
ด้วยเทคนิคการเรียนรู้ของเครื่อง



วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรมหาบัณฑิต  
สาขาวิชาวิศวกรรมอุตสาหการ ภาควิชาวิศวกรรมอุตสาหการ  
คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย  
ปีการศึกษา 2564  
ลิขสิทธิ์ของจุฬาลงกรณ์มหาวิทยาลัย

Thesis Title Analyzing NYPD Stop, Question, and Frisk with Machine Learning Techniques  
By Miss Passiri Bodhidatta  
Field of Study Industrial Engineering  
Thesis Advisor Associate Professor DARICHA SUTIVONG, Ph.D.

---

Accepted by the FACULTY OF ENGINEERING, Chulalongkorn University in  
Partial Fulfillment of the Requirement for the Master of Engineering

..... Dean of the FACULTY OF  
ENGINEERING  
(Professor SUPOT TEACHAVORASINSKUN, D.Eng.)

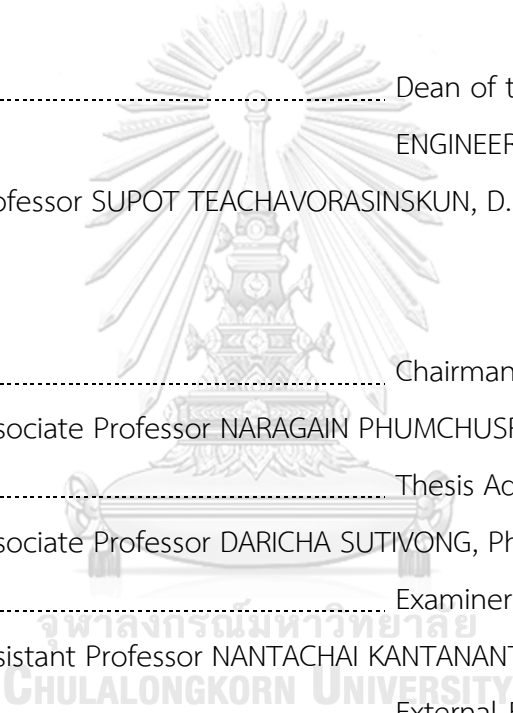
THEESIS COMMITTEE

..... Chairman  
(Associate Professor NARAGAIN PHUMCHUSRI, Ph.D.)

..... Thesis Advisor  
(Associate Professor DARICHA SUTIVONG, Ph.D.)

..... Examiner  
(Assistant Professor NANTACHAI KANTANANTHA, Ph.D.)

..... External Examiner  
(Associate Professor Chansiri Singhtaun, D.Eng.)



ภัสสิริ โพธิ์ทัด : การวิเคราะห์ปฏิบัติการเรียกสกัดจับสอบถามและค้นตัวของกรมตำรวจ  
นิวยอร์ก ด้วยเทคนิคการเรียนรู้ของเครื่อง. ( Analyzing NYPD Stop, Question, and  
Frisk with Machine Learning Techniques) อ.ที่ปรึกษาหลัก : รศ. ดร.ดาริชา สุธิ  
วงศ์

ถึงแม้ว่าการสกัดจับ ในปฏิบัติการเรียกสกัดจับ สอบถาม และค้นตัว ได้ลดลงอย่างมาก  
หลังจากการปฏิรูปกรมตำรวจนิวยอร์ก ในปี 2013 แต่การสกัดจับที่ไม่จำเป็น และการใช้อาวุธกับ  
ประชาชนผู้บริสุทธิ์ ยังคงเป็นปัญหาสำคัญ งานศึกษานี้ ได้วิเคราะห์การสกัดจับระหว่างปี 2014-  
2019 โดยใช้การเรียนรู้ด้วยเครื่องแบบต้นไม้ 3 ประเภท ได้แก่ Decision Tree, Random Forest  
และ XGBoost เพื่อสร้างแบบจำลองเพื่อทำนายการสกัดจับว่าจะมีการกระทำผิดหรือไม่ และเพื่อ  
ทำนายระดับการใช้กำลังของตำรวจ รวมทั้งระบุปัจจัยที่ส่งผล ผลการศึกษา แสดงให้เห็นว่า  
XGBoost ให้ผลลัพธ์ดีกว่าแบบจำลองอื่นในการทำนายทั้งสองปัญหา ในการทำนายความผิด ได้  
คะแนน F1 ที่ 65.9% และความแม่นยำ 84.0% ส่วนในการทำนายระดับการใช้กำลังของตำรวจ  
ได้คะแนน F1 ของระดับ 1 และระดับ 2 เป็น 40.7% และ 35.0% ตามลำดับ ด้วยความแม่นยำ  
โดยรวม 80.4% โดยผลลัพธ์ชี้ให้เห็นว่าการมีอาวุธสื่อถึงการที่ผู้ต้องสงสัยได้กระทำผิด ถึงกระนั้น  
ตำรวจอาจมีการสันนิษฐานที่ไม่แม่นยำเกี่ยวกับการครอบครองอาวุธของผู้ต้องสงสัย ซึ่งอาจนำไปสู่  
การสกัดจับ และการใช้ปืนกับประชาชนผู้บริสุทธิ์ได้ นอกจากนี้ งานศึกษานี้ยังได้ศึกษาเทคนิคการ  
ผสมผสานที่ชื่อว่า Super Learner โดยได้ทดลองสร้างโครงสร้างหลากหลายแบบ พบว่า Super  
Learner ให้ผลลัพธ์ที่พัฒนาขึ้นจากแบบจำลองพื้นฐานของมันเองเมื่อใช้แบบจำลองพื้นฐานที่ไม่ได้  
ปรับตั้งค่า แต่จะไม่พัฒนาขึ้นมากนักหากใช้แบบจำลองพื้นฐานที่ผ่านการปรับตั้งค่ามาแล้ว  
ความสามารถการทำนายของแบบจำลองพื้นฐานก็เป็นสิ่งหลักที่ส่งผลต่อความสามารถในการ  
ทำนายของ Super Learner เช่นกัน นั่นคือหากใช้แบบจำลองพื้นฐานที่มีความสามารถที่ดี ก็จะช่วย  
พัฒนาความสามารถของ meta model ได้ และในทางกลับกันก็เช่นกัน สุดท้ายพบว่า meta  
model ซึ่งใช้ XGBoost และ Logistic Regression ให้ผลลัพธ์ดีกว่าแบบจำลองอื่นในการทำ  
นายทั้ง 2 ปัญหา

สาขาวิชา วิศวกรรมอุตสาหการ

ลายมือชื่อนิสิต .....

ปีการศึกษา 2564

ลายมือชื่อ อ.ที่ปรึกษาหลัก .....

# # 6470247821 : MAJOR INDUSTRIAL ENGINEERING

KEYWORD: Stop Question and Frisk, NYPD, Tree - based machine learning, XGBoost, Super Learner

Passiri Bodhidatta : Analyzing NYPD Stop, Question, and Frisk with Machine Learning Techniques. Advisor: Assoc. Prof. DARICHA SUTIVONG, Ph.D.

Although stops from “Stop, Question, and Frisk” program have decreased dramatically after the New York Police Department (NYPD) reform in 2013, the unnecessary stops and weapon use against innocent citizens remain critical problems. This study analyzes the stops during 2014 – 2019, using three tree-based machine learning approaches: Decision Tree, Random Forest, and XGBoost. Models for predicting stops that resulted in a conviction and police’s level of force used are developed and driving factors are identified. Results show that XGBoost outperformed other models in both predictions. The performance of Guilty Prediction was at 65.9% F1 score and 84.0% accuracy. For Level of Force Prediction, the F1 score obtained for “Level 1” and “Level 2” were 40.7% and 35.0% respectively, with 80.4% overall accuracy. The findings indicated that the presence of a weapon implies a suspect's conviction. Despite that, numerous unnecessary stops are likely driven by inaccurate assumptions about suspect’s weapon possession, which lead to police’s gunfire usage against innocent citizens. Additionally, this study explores a hybrid technique called Super Learner. Experiments on various structures of Super Learners are performed. For base models, Super Learners can improve performance from their own base models when using untuned base models but do not improve when using tuned base models. The performance of base models also played a significant role in the performance of Super Learners, namely having high-performance base models improved meta models’ performance, and vice versa. For meta models, XGBoost and Logistic Regression outperform other meta models across both predictions.

Field of Study: Industrial Engineering

Student's Signature .....

Academic Year: 2021

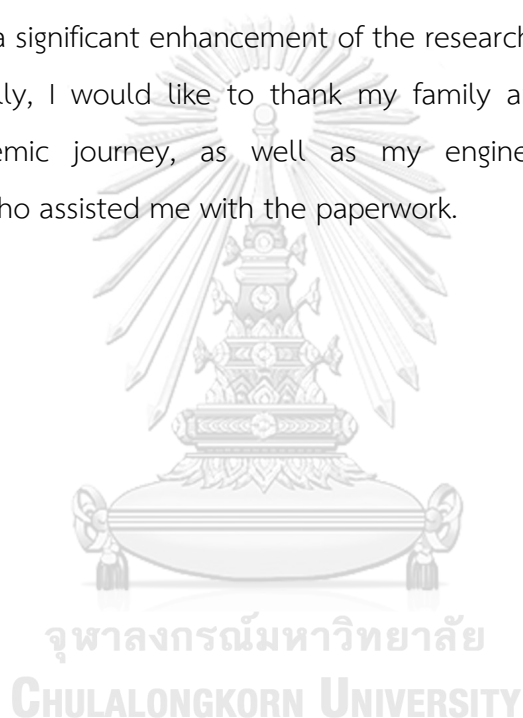
Advisor's Signature .....

## ACKNOWLEDGEMENTS

I would like to express my deepest gratitude to my advisor, Assoc. Prof. Daricha Sutivong. She had always kindly guided me through every step of the research since the final year of my bachelor's degree, including providing numerous incredible ideas and encouragement for the study from her expertise. I also would like to thank the thesis committee, Assoc. Prof. Naragain Phumchusri, Asst. Prof. Nantachai Kantanantha, and Assoc. Prof. Chansiri Singhtaun, for their constructive comments, which resulted in a significant enhancement of the research.

Additionally, I would like to thank my family and friends for their support during my academic journey, as well as my engineering senior, Mr. Thirawat Bannakulpiphat, who assisted me with the paperwork.

Passiri Bodhidatta



## TABLE OF CONTENTS

	Page
.....	iii
ABSTRACT (THAI).....	iii
.....	iv
ABSTRACT (ENGLISH).....	iv
ACKNOWLEDGEMENTS.....	v
TABLE OF CONTENTS.....	vi
LIST OF TABLES.....	ix
LIST OF FIGURES.....	xii
Chapter 1 : Introduction.....	1
1.1 Background.....	1
1.2 Objectives of the study.....	3
1.3 Scope of the study.....	3
1.4 Benefits of the study.....	4
Chapter 2 : Related Theories and Literature Review.....	5
2.1 Related Work.....	5
2.1.1 Studies analyzing NYPD Stop, Question, and Frisk datasets.....	5
2.1.2 Studies relating to police behavior using tree-based classification machine learning methods.....	6
2.1.3 Studies relating to the performance of tree-based classification techniques.....	8
2.1.4 Studies relating to Super Learner technique for improving the performance of established models.....	9

2.2 Related Theories .....	11
2.2.1 Classification models .....	11
2.2.1.1 Decision Tree.....	11
2.2.1.2 Random Forest, Extra Tree, and Balanced Random Forest .....	12
2.2.1.3 Gradient Boosting Decision Tree, XGBoost, AdaBoost, and Histogram-based Gradient Boosting Classification Tree. ....	13
2.2.1.4 Logistic Regression .....	15
2.2.1.5 Gaussian Naive Bayes and Multinomial Naïve Bayes .....	15
2.2.1.6 K Nearest Neighbor .....	16
2.2.1.7 Multilayer Perceptron Neural Network.....	17
2.2.2 Super Learner.....	19
2.2.3 Classification Metrics.....	20
2.2.4 SHAP (Shapley Additive eXplanation) value .....	23
Chapter 3 : Methodology.....	25
3.1 Problem Approach .....	25
3.2 Data source .....	25
3.3 Data Preparation and Output labeling .....	25
3.4 Exploratory Data Analysis.....	28
3.5 Tools and Model Construction .....	29
3.6 Evaluation metric.....	31
3.7 Super Learner experiment .....	32
Chapter 4 : Results and Discussion.....	33
4.1 Results of tree-based models.....	33
4.1.1 Guilty Prediction .....	33



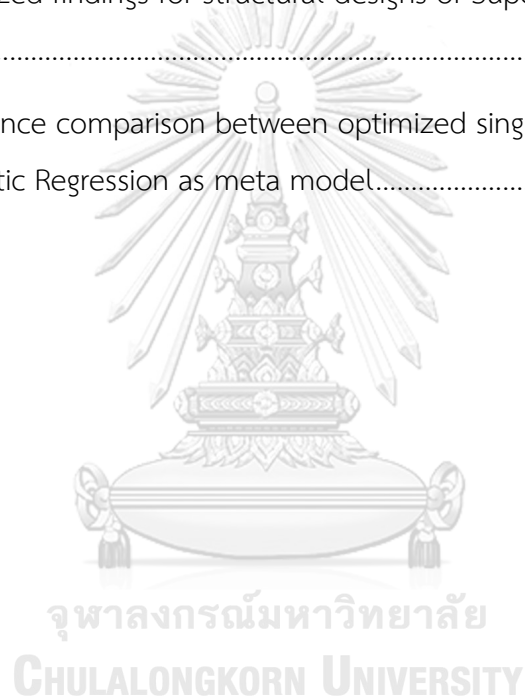
4.1.2 Level of Force Prediction.....	40
4.2 Discussion of tree-based models .....	47
4.2.1 Results interpretation .....	47
4.2.2 XGBoost Performance.....	48
4.2.3 Interpretation of important features' effect .....	48
4.2.3.1 Guilty Prediction .....	48
4.2.3.2 Level of Force Prediction .....	49
4.2.4 Appropriateness of the NYPD officer's behavior in SQF practices .....	51
4.2.5 Supplementary hypothesis testing among racial groups .....	52
4.3 Results of Super Learners .....	56
4.3.1 Guilty Prediction .....	56
4.3.2 Level of Force Prediction.....	62
4.3.3 Confusion Matrix observation .....	67
4.3.3.1 Guilty Prediction.....	67
4.3.3.2 Level of Force Prediction .....	68
4.4 Discussion of Super Learners.....	72
4.4.1 Base models .....	72
4.4.2 Meta models .....	73
4.4.3 Summarized findings for structural designs of Super Learners.....	74
Chapter 5 : Conclusion.....	77
REFERENCES .....	79
APPENDIX.....	84
VITA.....	99

## LIST OF TABLES

	<b>Page</b>
Table 1 Studies analyzing NYPD Stop, Question, and Frisk datasets .....	6
Table 2 Studies relating to police behavior using tree-based classification machine learning methods.....	7
Table 3 Studies relating to the performance of tree-based classification techniques..	8
Table 4 Studies relating to Super Learner technique for improving the performance of established models.....	10
Table 5 Confusion matrix.....	20
Table 6 Outcome description.....	28
Table 7 Super Learner experimental plan .....	32
Table 8 Average results from 4-fold cross-validation for Guilty Prediction.....	33
Table 9 Performance of the best XGBoost model on Test set for Guilty Prediction ..	34
Table 10 Top 10 important features for Guilty Prediction .....	37
Table 11 Effects toward Guilty class of important features the best XGBoost model	40
Table 12 Average results from 4-fold cross-validation for Level of Force Prediction..	40
Table 13 Performance of the best XGBoost model on Test set for Level of Force Prediction .....	41
Table 14 Top 10 important features for Level of Force Prediction.....	44
Table 15 Effects toward each Level of important features in the best XGBoost model .....	46
Table 16 Boroughs and Precinct numbers in New York City .....	51
Table 17 Results of hypothesis testing for “False stop” among racial groups .....	54
Table 18 Results of hypothesis testing for “Level of Force” among racial groups.....	55

Table 19 Average results from 4-fold cross-validation of the base models.....	57
Table 20 Average results from 4-fold cross-validation of the Super Learners .....	58
Table 21 Additional Super Learner experimental plan (Guilty Prediction) .....	59
Table 22 Results from 4-fold cross-validation of the base models (additional experiments).....	60
Table 23 Average results from 4-fold cross-validation of the Super Learners (additional experiments) .....	61
Table 24 Results from 4-fold cross-validation of the base models.....	62
Table 25 Results from 4-fold cross-validation of the Super Learners.....	63
Table 26 Additional Super Learner experimental plan (Level of Force Prediction).....	64
Table 27 Average result from 4-fold cross-validation of the base models (additional experiments).....	65
Table 28 Average results from 4-fold cross-validation of the Super Learners (additional experiments) .....	66
Table 29 Confusion matrix and related metrics of single XGBoost (untuned).....	67
Table 30 Confusion matrix and related metrics of single XGBoost (tuned).....	67
Table 31 Confusion matrix and related metrics of Super Learner in Experiment A1 (Untuned tree-based models as base models, Logistic Regression as meta model) ..	68
Table 32 Confusion matrix and related metrics of single XGBoost (untuned).....	69
Table 33 Confusion matrix and related metrics of single XGBoost (tuned).....	69
Table 34 Confusion matrix and related metrics of Super Learner in Experiment B1 (untuned tree-based models as base models, XGBoost as meta model) .....	70
Table 35 Confusion matrix and related metrics of Super Learner in Experiment B3 (untuned tree-based, Logistic Regression, and Gaussian Naïve Bayes as base models, Random Forest as meta model).....	70

Table 36 Confusion matrix and related metrics of Super Learner in Experiment B2 (tuned tree-based models as base models, XGBoost as meta model).....	71
Table 37 Confusion matrix and related metrics of Super Learner in Experiment B4 (tuned tree-based, Logistic Regression, and Gaussian Naïve Bayes as base models, Logistic Regression as meta model).....	71
Table 38 Summarized findings for structural designs of Super Learners (base models) .....	74
Table 39 Summarized findings for structural designs of Super Learners (meta models) .....	75
Table 40 Performance comparison between optimized single XGBoost and Super Learner with Logistic Regression as meta model.....	76



## LIST OF FIGURES

	Page
Figure 1 The number of stops by NYPD in 2011-2019.....	2
Figure 2 The percentage of stops without arrest or summon issued.....	2
Figure 3 The percentage of innocent suspects stopped with police’s use of weapons2	2
Figure 4 Example of a decision tree for a training set (Kotsiantis, 2013). .....	12
Figure 5 Voronoi tessellation (Peterson, 2009).....	17
Figure 6 Example structure of a Multilayer Perceptron Neural Network with 2 hidden layers (Haykin, 2009) .....	18
Figure 7 Example structure of Super Learner algorithm (Neto et al., 2020) .....	19
Figure 8 Example of ROC curve and AUC (Fawcett, 2006).....	22
Figure 9 ROC and Precision-Recall Curve under skewed data (Fawcett, 2006).....	23
Figure 10 The number of observations for each “Guilty” class .....	28
Figure 11 The number of observations in each “Level of force” class .....	29
Figure 12 Precision-Recall Curves for Guilty Prediction.....	34
Figure 13 Important features in the best Decision Tree model for Guilty Prediction..	35
Figure 14 Important features in the best Random Forest model for Guilty Prediction .....	35
Figure 15 Important features in the best XGBoost model for Guilty Prediction .....	36
Figure 16 Effects of important features toward Guilty class based on SHAP value .....	38
Figure 17 SHAP value of “Y coordinate”, “X coordinate”, “Precinct” and “Suspect’s age” .....	39
Figure 18 Important features in the best Decision Tree model for Level of Force Prediction .....	42

Figure 19 Important features in the best Random Forest model for Level of Force Prediction based on average magnitude of SHAP values (Class = Level) .....	42
Figure 20 Important features in the best XGBoost model for Level of Force Prediction .....	43
Figure 21 Effects of important features toward each Level based on SHAP value.....	45
Figure 22 SHAP values toward Level 1 for “Suspect’s age” .....	47
Figure 23 Plots of SHAP value toward each Level for Precinct .....	51
Figure 24 The number of observations for each racial group.....	53
Figure 25 Percentages of “False stop” for each racial group.....	53
Figure 26 Percentages of “Level of Force” for each racial group.....	55



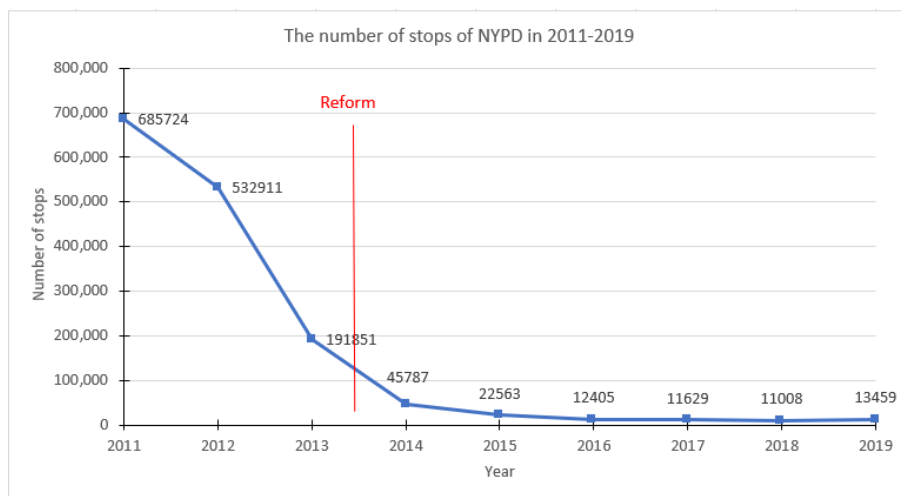
# Chapter 1 : Introduction

## 1.1 Background

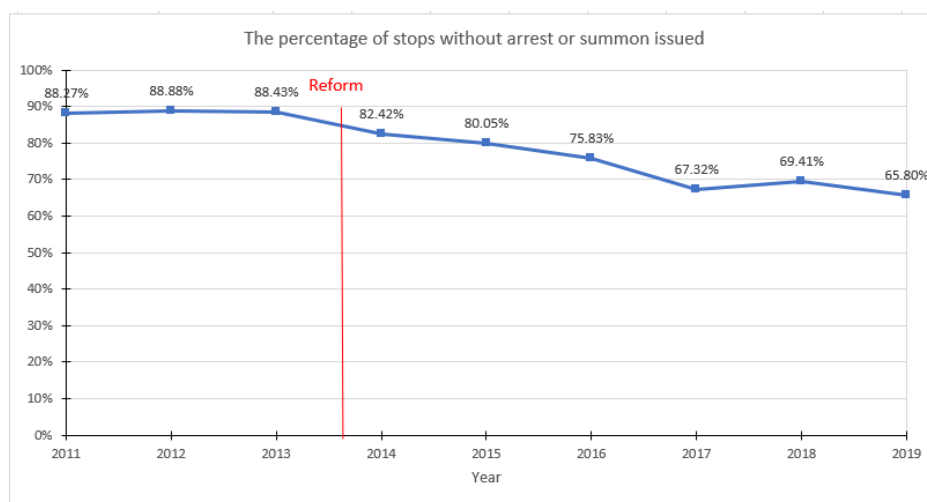
The ‘Stop, Question, and Frisk (SQF)’ program in New York City was established from the decision of Terry v. Ohio by The United States Supreme Court in 1968. A Terry Stop is a lawful practice that allows the United States officers to stop and temporarily detain suspects when there is a reasonable suspicion of criminal involvement. The officers are also allowed to pat down the suspect’s clothing in order to search for weapons and contrabands, known as ‘Frisk’. Arresting and issuing a search warrant may proceed if probable cause is found. When a police officer stops a suspect on a street or in a vehicle, it is called ‘Stop and Frisk’ and ‘Vehicle Stop’ respectively.

Nowadays, U.S. police officers are questioned about their biased behaviors towards suspects, including those in New York City, where the population is highly diverse. The New York Police Department (NYPD) has a high number of officers per capita among others in the United States. Despite the benefit of law enforcement, unnecessary stops and excessive actions were frequently mentioned in recent years.

From the statistical record of stops published, the number of stops rapidly increased, from 97,296 stops in 2002 to its peak at 685,724 stops in 2011. Since its peak in 2011, the number of stops has significantly dropped as shown in Figure 1. Although the proportion of stops without arrest or summon issued has also dropped after the reform as shown in Figure 2, it is still high (76.6% of all stops after the reform) compared to those that resulted in a conviction (23.4% of all stops after the reform).

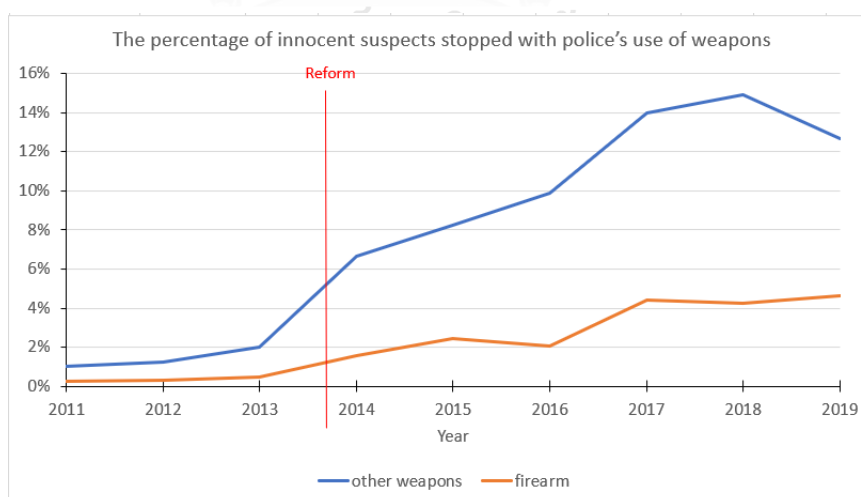


**Figure 1** The number of stops by NYPD in 2011-2019



**Figure 2** The percentage of stops without arrest or summon issued

During the Stop, Question, and Frisk practices, the officers may use physical forces against suspects in various ways, with or without weapons, such as handcuffs, baton, conducted energy weapon (CEW), and firearm. Although it is legal to use weapons during SQF practices, a suspect who was not convicted should not encounter any forces. After the reform, 12.21% of innocent suspects were stopped with police's use of weapons, and the percentage tends to increase as shown in Figure 3.



**Figure 3** The percentage of innocent suspects stopped with police's use of weapons



As demonstrated, despite the NYPD reform, the unnecessary stops and physical force with weapons used towards innocent citizens by officers still are critical problems. To investigate the issue, this study will analyze the SQF dataset during 2014 – 2019 retrieved from the NYPD website. With data analytic techniques, driven factors for stops which resulted in a conviction and police's weapons usage can be found. These factors can be analyzed whether an officer used weapons based on a sign of guilty suspect or suspect's other characteristics. Furthermore, the discovered insights can be applied to enhance law enforcement in New York City.

Additionally, with a lot of social issues that have been continuously gaining public's interest in recent years, exploring this issue is the opportunity to extend the usability of data analytics, especially machine learning methods. It is also crucial to analyze this area with quantitative reference besides other established areas, such as finance and healthcare. However, lack of data collection was the main obstacle for the social area. The SQF dataset of New York City is, fortunately, one of the very detailed police operation dataset published. Taken together, analysis of this dataset using machine learning methods was the chosen topic to experiment in this project.

## 1.2 Objectives of the study

1. To create a model and investigate the factors relating to an arrest or a summon issued after a stop, using classification machine learning techniques.
2. To create a model and investigate the factors relating to police's weapons usage during a stop, using classification machine learning techniques.
3. To explore a hybrid approach aiming to enhance the performance of the established models.

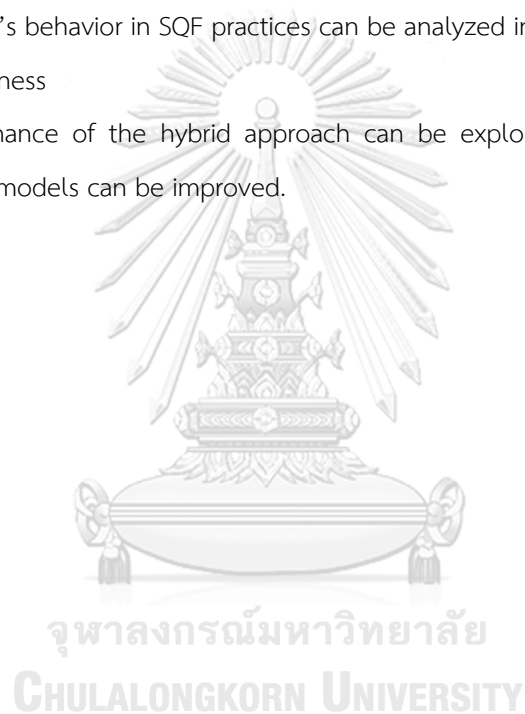
## 1.3 Scope of the study

1. This study analyzed the Stop, Question, and Frisk datasets during 2014 – 2019, published on the NYPD website.
2. The model creation was based on stopped suspects during 2014 – 2019 as population, which did not include all of New York City citizens.
3. The study only explored the factors contained in the datasets.
4. The investigated machine learning techniques were tree-based classification, such as Decision Tree, Random Forest, and XGBoost.

5. The hybrid technique, Super Learner, was explored. It was composed of various base learners such as Decision Tree, Random Forest, XGBoost, Logistic Regression, and Gaussian Naive Bayes, with a meta learner such as Logistic Regression, Decision Tree, Random Forest, XGBoost, and Neural Network.

#### 1.4 Benefits of the study

1. The driven factors for an arrest or a summon issued after a stop and police's weapons usage during a stop can be identified.
2. NYPD officer's behavior in SQF practices can be analyzed in terms of its appropriateness
3. The performance of the hybrid approach can be explored, and the performance of established models can be improved.



## Chapter 2 : Related Theories and Literature Review

### 2.1 Related Work

#### 2.1.1 Studies analyzing NYPD Stop, Question, and Frisk datasets

The United States officers' behavior has been an interesting issue in society as well as researchers for many years. Terry Stop or Stop and Frisk practices in many states were also investigated. In recent years, many researchers have focused on analyzing the Stop, Question, and Frisk practices of NYPD officers by using quantitative methods.

The racial disparity in SQF practices before the NYPD reform was investigated by some researchers. A study from SQF records during 1998-1999, using Poisson regression models, showed that African and Hispanic pedestrians were stopped more than whites after controlling for racial population variability and crime rate across precincts (Gelman et al., 2007). However, a study conducted later which used SQF data during 2003-2011 with precinct-level fixed regression models, indicated that race was not a significant factor for deciding whom to stop (Coviello & Persico, 2015).

The consequence of the NYPD reform was also examined by a study using external and internal benchmarking approach, with indicators measuring whether the stops resulted in frisk, search, summon issued, arrest, and use of force. The study showed that during 2013-2015 after the reform, race was no longer a significant factor, even though there was some racial disparity in 2012 before the reform (MacDonald & Braga, 2019).

In addition to ethnics, several studies suggested some relationships between SQF practices in New York City and other suspect characteristics. Using the generalized linear mixed models to analyze SQF data during 2006-2013, a study showed that among male suspects who were above 18 years old, Black and Hispanic men with large BMI (body mass index) were more likely to be frisked, searched, or encountered physical force (Milner et al., 2016). Some situational characteristics, such as a suspect was proximal to the scene, had significant relationships to physical force usage as well. The evidence was shown in another study using SQF data in 2012 with logistic regression models (Morrow et al., 2017). The summary of this topic is shown in Table 1.

**Table 1** Studies analyzing NYPD Stop, Question, and Frisk datasets

Researchers	Data used	Techniques	Results
Gelman et al., 2007	SQF records during 1998-1999	Poisson regression models, controlling racial population variability and crime rate across precincts	African and Hispanic pedestrians were stopped more than whites.
Coviello & Persico, 2015	SQF records during 2003-2011	Precinct-level fixed regression models	Race was not a significant factor for deciding whom to stop.
MacDonald & Braga, 2019	SQF records during 2012-2015	External and internal benchmarking approach	After the reform, race was no longer a significant factor measuring whether the stops resulted in frisk, search, summon issued, arrest, and use of force, even though there was some racial disparity in 2012 before the reform.
Milner et al., 2016	SQF records during 2006-2013	Generalized linear mixed models	Among male suspects who were above 18 years old, Black and Hispanic men with large BMI (body mass index) were more likely to be frisked, searched, or encountered physical force.
Morrow et al., 2017	SQF records during 2012	Logistic regression models	Some situational characteristics, such as a suspect was proximal to the scene, had significant relationships to physical force usage.

### 2.1.2 Studies relating to police behavior using tree-based classification machine learning methods.

Apart from the NYPD SQF dataset, tree-based machine learning classifiers were used in several research papers that focused on the U.S. police law enforcement. Public perception of

police behavior during traffic stops was predicted in a study, using Random Forest classification and conventional logistic regression. The results of both approaches were consistent in terms of a principal factor, which was drivers' belief that a stop is legal (Hu et al., 2021). Another study, aiming to predict the U.S. police adverse incidents, compared the performance among tree-based classifiers. The result showed that Extra Tree, similar to Random Forest, had the best performance among others (Helsby et al., 2017).

Police law enforcement outside the United States was also investigated. Supervised machine learning techniques, including Decision Tree and Random Forest, were used to predict re-arrest by police in Santiago, Chile. The prediction was conducted based on data regarding to previous arrests and personal information, such as gender and age. The study showed that all models had achieved excellent performance (van 't Wout et al., 2021). The summary of this topic is shown in Table 2.

**Table 2** Studies relating to police behavior using tree-based classification machine learning methods.

Researchers	Data used	Techniques	Results
Hu et al., 2021	Police traffic stops surveys in 2005, 2008, 2011, and 2015, from BJS police–public contact surveys (PPCS)	Random Forest classification and Logistic Regression	Public perception of police behavior during traffic stops was predicted. The results of both approaches were consistent in terms of a principal factor.
Helsby et al., 2017	Datasets from Charlotte-Mecklenburg Police Department	Extra Tree, Random Forest, Logistic Regression, and Ada Boost	Extra Tree, similar to Random Forest, had the best performance among others (AUC = 0.67).
van 't Wout et al., 2021	Arrests history in Santiago de Chile and personal metadata, such as age and gender	Decision Tree, Random Forest, Logistic Regression, Naïve Bayes, Multilayer Perceptron	All models had achieved excellent performance (AUC = 0.81 for all models).

### 2.1.3 Studies relating to the performance of tree-based classification techniques.

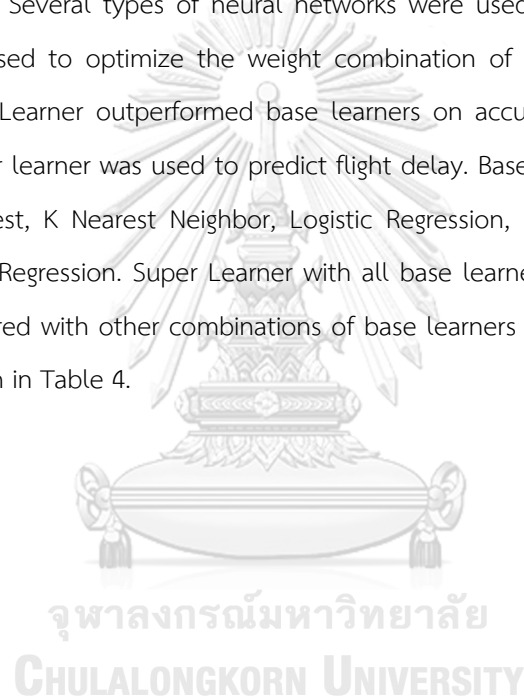
Tree-based classification techniques were applied and compared by many researchers in various fields. A study regarding to star and galaxy classification with a photometric dataset was carried out. They used many classification algorithms including Function Tree, Random Forest, Gradient Boosting Decision Tree (GBDT), Adaboost, and XGBoost (eXtrem Gradient Boosting). The result showed that XGBoost outperformed other models (Chao et al., 2019). In another study, cardiovascular disease was investigated by analyzing data with machine learning techniques such as Random Forest, GBDT, and XGBoost. From experimental results, XGBoost had the best performance among other models (Jiang et al., 2021). XGBoost also outperformed Random Forest in a study predicting water table depth, for improving agricultural production efficiency (Brédy et al., 2020). The summary of this topic is shown in Table 3.

**Table 3** Studies relating to the performance of tree-based classification techniques

Researchers	Data used	Techniques	Results
Chao et al., 2019	Photometric data set from Sloan Digital Sky Survey-DR7.	Function tree (FT), Adaptive boosting (Adaboost), Random Forest (RF), Gradient Boosting Decision Tree (GBDT), Stacked Denoising AutoEncoders (SDAE), and Deep Belief Nets (DBN)	XGBoost outperformed other models (accuracy 99.87% for bright stellar, 95.72% for dark stella, and 79.48% for the darkest stellar).
Jiang et al., 2021	Data from patients with suspected cardiovascular disease presenting at ED triage.	XGBoost, Gradient Boosting Decision Tree, Random Forest, Multinomial Logistic Regression.	XGBoost had the best performance among other models (AUC = 0.937)
Brédy et al., 2020	Water table depth measured from a cranberry farm, near Québec City, Québec, Canada	Random Forest and XGBoost.	XGBoost outperformed Random Forest, in terms of RMSE and NSE.

#### 2.1.4 Studies relating to Super Learner technique for improving the performance of established models.

To improve the performance of tree-based classifiers, Super Learner technique, which combined various types of classifiers was proposed and explored by many researchers. Transient prediction of CO<sub>2</sub> and NO<sub>x</sub> of diesel trucks was conducted in a study, using tree-based classifiers including Random Forest, XGBoost, LightGBM, and CatBoost as base learners for Super Learner. The result showed that the Super Learner model outperformed the traditional method (Wei et al., 2022). A Super Learner ensemble was also proposed for the vehicle-type image classification problem in a study. Several types of neural networks were used as base learners. The Super Learner was then used to optimize the weight combination of the base learners. The result showed that Super Learner outperformed base learners on accuracy (Hedeya et al., 2020). In another study, Super learner was used to predict flight delay. Base learners were Gaussian Naïve Bayes, Random Forest, K Nearest Neighbor, Logistic Regression, and Decision Tree, and Meta learner was Logistic Regression. Super Learner with all base learners outperformed on accuracy and f1 score compared with other combinations of base learners (Yi et al., 2021). The summary of this topic is shown in Table 4.



**Table 4** Studies relating to Super Learner technique for improving the performance of established models.

Researchers	Data used	Techniques	Results
Wei et al., 2022	Onboard test data of 9 China VI N2 vehicles	Tree-based classifiers including Random Forest, XGBoost, LightGBM, and CatBoost as base learners for Super Learner.	Super Learner model outperformed the traditional method
Hedeya et al., 2020	The MIOvision Traffic Camera Dataset (MIO-TCD) and the Beijing Institute of Technology's (BIT) vehicle classification dataset.	Several types of neural networks were used as base learners. The Super Learner was used to optimize the weight combination of the base learners.	Super Learner outperformed base learners on accuracy
Yi et al., 2021	Flight data from January to December 2019 at Logan International Airport in Boston, Massachusetts, the United States.	Super Learner. Base learners were Gaussian Naïve Bayes, Random Forest, K Nearest Neighbor, Logistic Regression, and Decision Tree, and Meta learner was Logistic Regression	Super Learner with all base learners outperformed on accuracy and f1 score compared with other combinations of base learners

According to all studies mentioned above, it suggests that there are possible relationships between suspects' or situational characteristics and SQF practices. However, the SQF datasets in the last few years, after the reform, are not much explored. Moreover, most methods used in previous works with the NYPD SQF datasets are statistical methods, while machine learning models seem to be effective tools used by researchers in relevant fields. Tree-based algorithms such as Decision Tree and Random Forest showed great performance in related predictive problems. Also, according to the previous section, XGBoost is an algorithm that outperformed Random Forest in many studies. Since this study aims to find related factors to the practices, it examines the NYPD SQF dataset in the last few years with tree-based machine



learning algorithms including Decision Tree, Random Forest, and XGBoost to create predictive models. With these models, the driven factors for probable cause that leads to an arrest or issuing a summon, and a decision on weapons used, can be found.

Additionally, this study explores the potential hybrid approach, Super Learner, on top of the established models previously mentioned, to seek opportunities for enhancing the prediction performance. For base learners, Decision Tree, Random Forest, and XGBoost are chosen from a past study which showed a good performance of Super Learner with tree-based models as base models (Wei et al., 2022). In order to explore other predictive models with different structures, Logistic Regression and Gaussian Naïve Bayes are used with tree-based models, inspired by a past study that showed Super Learner with various base learners, including Decision Tree, Logistic Regression, and Gaussian Naïve Bayes, performed well on accuracy and f1 score (Yi et al., 2021). For meta learners, Logistic Regression is chosen from its good performance in the past study (Yi et al., 2021). A neural network algorithm, Multilayer Perceptron (MLP), is also chosen inspired by the past study using neural networks for Super Learner (Hedeya et al., 2020). However, due to heavy computation, this study only uses MLP as a meta learner and not as base learners. Lastly, since tree-based models are not much explored as meta learners in past studies, this study uses Decision Tree, Random Forest, and XGBoost to investigate how well tree-based models perform as meta learners.

Besides different techniques, this study also explores how the optimization of base models affects Super Learner, by using both tuned and untuned base models. Since the Super Learner consists of many models and training processes, using untuned base models may have benefits in terms of efficient resource usage, and this kind of experiment is not much explored in past studies.

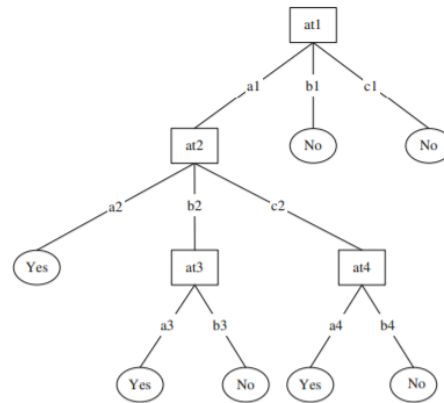
## 2.2 Related Theories

### 2.2.1 Classification models

#### 2.2.1.1 Decision Tree

Decision Tree is one of the supervised learning algorithms, for both regression and classification problems. The model is relating to recursively partition or segment attributes space into several regions. Predictors' value is compared to a threshold for numerical predictors or a set of values for categorical predictors. Decision Tree is commonly used because its model is

understandable, which is beneficial for interpretation. An example of decision tree analyzed from a training set is shown in Figure 4 (Kotsiantis, 2013; Lin et al., 2006).



**Fig. 1** A decision tree

**Table 1** Training set

at1	at2	at3	at4	Class
a1	a2	a3	a4	Yes
a1	a2	a3	b4	Yes
a1	b2	a3	a4	Yes
a1	b2	b3	b4	No
a1	c2	a3	a4	Yes
a1	c2	a3	b4	No
b1	b2	b3	b4	No
c1	b2	b3	b4	No

**Figure 4** Example of a decision tree for a training set (Kotsiantis, 2013).

In classification problems, the predicted class for each record is the most frequently occurring class for the region in which the record comes under. Splitting each node can be impure if not all training observations in divided regions fall into the same class. The process of partitioning or splitting nodes is commonly measured its purity by two metrics, Gini Index and Entropy. Gini index can be written as Equation 1, and Entropy as Equation 2 (Lin et al., 2006).

$$G = \sum_{k=1}^K \hat{p}_{mk}(1 - \hat{p}_{mk}) \quad (1)$$

$$D = - \sum_{k=1}^K \hat{p}_{mk} \log \hat{p}_{mk} \quad (2)$$

Where  $0 \leq \hat{p}_{mk} \leq 1$  is the proportion of  $k^{\text{th}}$  class training observations that fall into the  $m^{\text{th}}$  region. The smaller value of both metrics refers to more node purity.

### 2.2.1.2 Random Forest, Extra Tree, and Balanced Random Forest

Random forest is an ensemble of decision trees trained with bagging method. Bootstrap aggregation or bagging is a process of decreasing variance by building separate models from sets

of observations sampled from the dataset. The result is the average of all predictions as shown in Equation 3.

$$\hat{f}_{bag}(x) = \frac{1}{B} \sum_{b=1}^B \hat{f}^{*b}(x) \quad (3)$$

Where  $\hat{f}^{*b}(x)$  is the result from each  $b^{\text{th}}$  bootstrapped training set.

For classification problems,  $\hat{f}_{bag}(x)$  is obtained from the most common result from all models. Since the Decision Tree model mentioned earlier mostly results in high variance, the Bagging method helps improve the algorithm. However, all trees from the traditional bagging method are mostly similar, with the same strong predictors in the top nodes. Random forest increases randomness in the traditional bagging method by instead of selecting the most important attributes while growing the trees, it selects from a random subset of all attributes (Lin et al., 2006).

Extra Tree is another decision tree ensemble approach. When splitting, the features and cut-off values are randomized in each node. Extra Tree has the benefit of being efficient, and it can be improved by tuning hyperparameters for each problem (Geurts et al., 2006).

Balanced Random Forest is an adaptation of Random Forest. The training set for each tree is modified for class imbalance. It is built by combining two bootstrapped data sets, the minority class and the majority class. Both have the size of the minority class (Kobyliński & Przepiórkowski, 2008).

### 2.2.1.3 Gradient Boosting Decision Tree, XGBoost, AdaBoost, and Histogram-based Gradient Boosting Classification Tree.

Gradient Boosting Decision Tree (GBDT) is an algorithm based on aggregated Decision Trees as shown in Equation 4.

$$f_M(x) = \sum_{m=1}^M T(x; \theta_m) \quad (4)$$

Where  $x$  is the sample dataset

$T(x; \theta_M)$  is the Decision Tree

$\theta_M$  is the parameters of the Decision Tree

$M$  is the number of Decision Trees

The model is sequentially built by previous trees, the  $m^{\text{th}}$  step model is shown in Equation 5.

$$f_m(x) = f_{m-1}(x) + T(x; \theta_m) \quad (5)$$

Where  $f_{m-1}(x)$  is the present model, the next model parameters or  $\hat{\theta}_m$  is calculated from minimizing loss function as shown in Equation 6.

$$\hat{\theta}_m = \arg \min_{\theta_m} \sum_{i=1}^N L(y_i, f_{(m-1)}(x) + T(x; \theta_m)) \quad (6)$$

Where  $y_i$  is true class of the  $i^{\text{th}}$  observation. The  $L$  function is a loss function which depends on the type of the problem, negative binomial log-likelihood may be used for classification problems as shown in Equation 7-9 (Jerome, 2001)

$$L(y, F) = \log(1 + \exp(-2yF)) \quad (7)$$

$$\text{Where } F(x) = \frac{1}{2} \log \left[ \frac{\Pr(y = 1|x)}{\Pr(y = -1|x)} \right] \quad (8)$$

And the pseudorespond (Pr) is

$$\tilde{y}_i = 2y_i / (1 + \exp(2y_i F_{m-1}(x_i))) \quad (9)$$

XGBoost is different from Gradient Boosting Decision Tree by adding second-order Taylor expansion for optimizing the loss function, while the first-order derivative information obtained in GBDT is still preserved. This technique makes the model converge quicker. Furthermore, a regular term is also added in the loss function to prevent the model from overfitting (Chen & Guestrin, 2016)

Another Boost algorithm is AdaBoost. AdaBoost is the algorithm that trains the weak learners (in this case, decision trees) by starting from using original dataset. Then sequentially trains other same weak learners with weight-adjusting datasets, focusing more on the wrongly predicted samples (Freund & Schapire, 1997).

Lastly, Histogram-based Gradient Boosting Classification Tree is the implementation of GBDT based on LightGBM (Ke et al., 2017). The histogram-based models are significantly faster than GBDT when the dataset is large. The splitting nodes are greatly decreased by separating data into integer bins, as in a histogram. The algorithm also supports missing values (Pedregosa et al., 2011).

#### 2.2.1.4 Logistic Regression

Logistic Regression is a model which is widely used for classification problems, it is defined as the conditional probability shown in Equation 10.

$$P(y_i = \pm 1 | x_i) = \pi_i = \frac{1}{1 + e^{-x_i \beta}} \quad (10)$$

Where  $x_i$ ,  $i = 1, 2, 3, \dots, n$  is a sample from  $n$  observations with the vectors of independent variables which have  $(1 \times k)$  dimensions.

$y_i \in \{0, 1\}$ ,  $i = 1, 2, 3, \dots, n$  is a class label for the  $i^{th}$  observation

The maximum likelihood estimates for regression parameters,  $\beta_r$ ,  $r = 1, 2, \dots, k$ , can be obtained by the score equation as shown in Equation 11.

$$\frac{\partial \log L}{\partial \log B_r} = U(\beta_r) = \sum_{i=1}^n (y_i - \pi_i) x_i = 0 \quad (11)$$

Where  $L$  is a likelihood function.

To generate finite estimates while minimizing bias, maximizing Equation 12 is recommended (Firth, 1993).

$$\log L(\beta)^* = \log L(\beta) + 0.5 \log |I(\beta)| \quad (12)$$

Where  $0.5 \log |I(\beta)|$  is a penalty function (Bacaksiz & Koç, 2021; Ecevit, 2008; Yu et al., 2011).

#### 2.2.1.5 Gaussian Naive Bayes and Multinomial Naive Bayes

Naive Bayes is a supervised learning algorithm based on Bayes' theorem. The algorithm is popular because of its simplicity, making it easier to construct and more effective. According to Bayes' theorem, the probability of an instant  $E$  being in a class  $C$  is shown in Equation 13.

$$p(c|E) = \frac{p(E|c)p(c)}{p(E)} \quad (13)$$

$E$  is classified as class  $C = +$  (positive class) only if Equation 14 is satisfied.

$$f_b(E) = \frac{p(C = +|E)}{p(C = -|E)} \geq 1 \quad (14)$$

Where  $f_b(E)$  is called a Bayesian Classifier.

Given an instant  $E$  is represented by the features  $x_1, \dots, x_n$ . Naïve Bayesian is assumed that all features are independent given class  $C$ , that is shown in Equation 15.

$$p(E|C) = p(x_1, \dots, x_n|C) = \prod_{i=1}^n p(x_i|C) \quad (15)$$

Then a Naïve Bayesian classifier  $f_{nb}(E)$  is shown as Equation 16 (Zhang, 2004).

$$f_{nb}(E) = \frac{p(C=+)}{p(C=-)} \prod_{i=1}^n \frac{p(x_i|C=+)}{p(x_i|C=-)} \quad (16)$$

For Gaussian Naïve Bayesian algorithm, when a feature has continuous values, the probability of the value is assumed to be under Gaussian Distribution, defined by Equation 17.

$$P(x_i|C) = g(x, \mu, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \quad (17)$$

Where  $\mu$  is a mean and  $\sigma$  is a standard deviation (Han et al., 2011).

For Multinomial Naïve Bayesian algorithm, which is suitable for discrete features and widely used in text classification, the probability of a text given class  $C$  is shown in Equation 18.

$$P(t_i|C) = (\sum_n f_{ni})! \prod_n \frac{P(w_n|c)^{f_{ni}}}{f_{ni}!} \quad (18)$$

Where  $f_{ni}$  is the count of word  $w_n$  in the size of word vocabulary  $N$  and  $P(w_n|c)$  is the probability of  $w_n$  given class  $C$  (Kibriya et al., 2004).

#### 2.2.1.6 K Nearest Neighbor

K Nearest Neighbor classification is another one of the basic and simple classification algorithms. It is recommended when the distribution of data is unknown or has not been determined.

The Euclidean distance between a test sample and the required training samples is widely used to train the K Nearest Neighbor classifier. The Euclidean distance between the sample  $x_i$  ( $i = 1, 2, 3, \dots, n$ ) and  $x_l$  ( $l = 1, 2, 3, \dots, n$ ) is shown in Equation 19.

$$d(x_i, x_l) = \sqrt{(x_{i1} - x_{l1})^2 + (x_{i2} - x_{l2})^2 + \dots + (x_{ip} - x_{lp})^2} \quad (19)$$

Where each  $x_i$  has  $p$  features  $(x_{i1}, x_{i2}, \dots, x_{ip})$

The nearest neighbor is based on the Voronoi tessellation concept as shown in Figure 5. The Voronoi cells surround all 19 “+” marks, which are 19 samples. Each Voronoi cell,  $R_i$ , contains all of the surrounding points that are closest to each sample  $x_i$ , as defined in Equation 20.

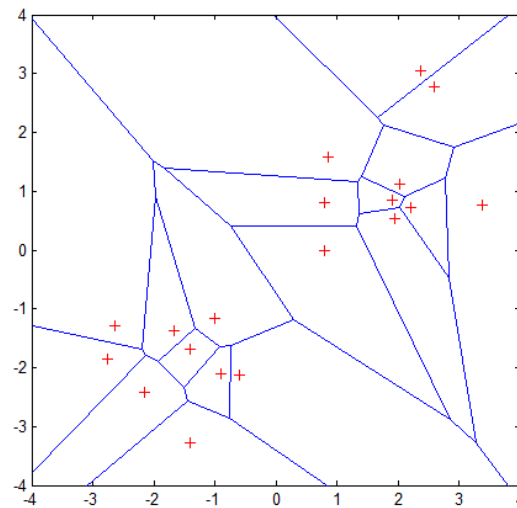


Figure 5 Voronoi tessellation (Peterson, 2009).

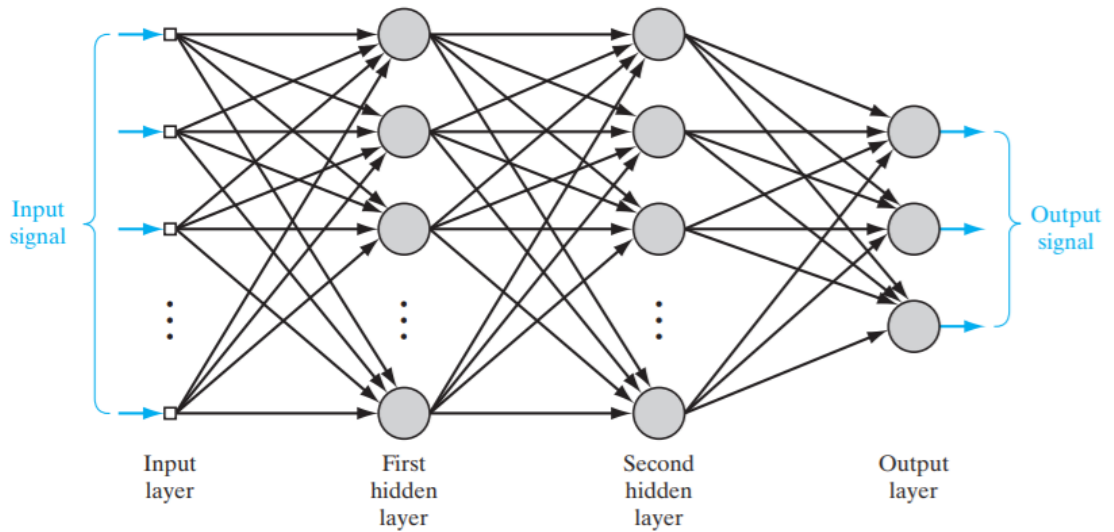
$$R_i = \{x \in R_p : d(x, x_i) \leq d(x, x_m), \forall i \neq m\} \quad (20)$$

Where  $x$  is all possible points within Voronoi cell  $R_i$  (Peterson, 2009).

#### 2.2.1.7 Multilayer Perceptron Neural Network

Artificial Neural Network (ANNs) has been driven from the concept of the way a human brain works, which is different from a normal digital computer. Because it is highly complex, nonlinear, and can work in parallel, the brain is capable of specific calculations by organizing its structural units, called neurons.

Multilayer Perceptron is a structure of neural networks which consists of hidden layers besides the input and output layer. Each neuron in the network has a “differentiable” nonlinear activation function and the network is highly connected. These hidden layers detect features that characterize the data. Figure 6 shows a Multilayer Perceptron Neural Network that is fully connected, which means each neuron is connected to all neurons in the former layer (Haykin, 2009).



**Figure 6** Example structure of a Multilayer Perceptron Neural Network with 2 hidden layers  
(Haykin, 2009)

Given a network that has one hidden layer, the inputs  $x_i$  are multiplied by their weight  $w_{ij}$  to generate preactivation functions  $y_j$  for each neuron.  $y_j$  are input into the non-linear activation function  $f_j$  in the hidden layer to generate outputs  $h_j$ . The preactivation functions of output neurons  $y_k$  are also generated with all  $h_j$  multiplied by their weight  $w_{jk}$ . The final output  $p_k$  is calculated using linear activation function  $f_k$ . The computation is shown in Equation 21 – 24.

$$y_j = b_j + \sum_i x_i w_{ij} \quad (21)$$

$$h_j = f_j(b_j + \sum_i x_i w_{ij}) \quad (22)$$

$$y_k = b_k + \sum_j f_j(b_j + \sum_i x_i w_{ij}) w_{jk} \quad (23)$$

$$p_k = f_k(b_k + \sum_j f_j(b_j + \sum_i x_i w_{ij}) w_{jk}) \quad (24)$$

Where  $b_j, b_k$  are biases of neurons in hidden and output layers. The error function is defined from the difference of the outputs and the expected results as shown in Equation 25.

$$E = \frac{1}{2} \sum_k (t_k - p_k)^2 \quad (25)$$

Where  $t_k$  is the expected result.



The basic training method of the network is the Back-Propagation algorithm. This technique is used to identify the effect of weights on the output prediction and to change the obtained weights for error reduction (Agustika et al., 2021; Akbaş & Özdemir, 2020).

## 2.2.2 Super Learner

Super Learner is an ensemble algorithm based on the idea of stacking, which is the process of combining outputs from trained models together for the final prediction (Wolpert, 1992). The Super Learner algorithm involves combining results of various learners and uses cross-validation to select the learners among the candidates. To compute the best ensemble weight vector, it finds the optimal combination of the base learners by minimizing the  $v$ -fold cross-validation loss. (Laan et al., 2007; Polley & Laan, 2010; van 't Wout et al., 2021).

The structure of Super Learner is shown in Figure 7. The training dataset is divided into  $V$  blocks, each block is trained with base learners, then the validation set of each block is predicted. After that, predictions from all base learners of every block are combined as the input for training the meta model. Finally, evaluation is made by training each base learner with the entire train dataset and then predicting the validation dataset. The meta learner then also uses the predictions of all base learners to predict the validation set. Performance on validation set between each base learner and the Super Learner can be compared (Neto et al., 2020).

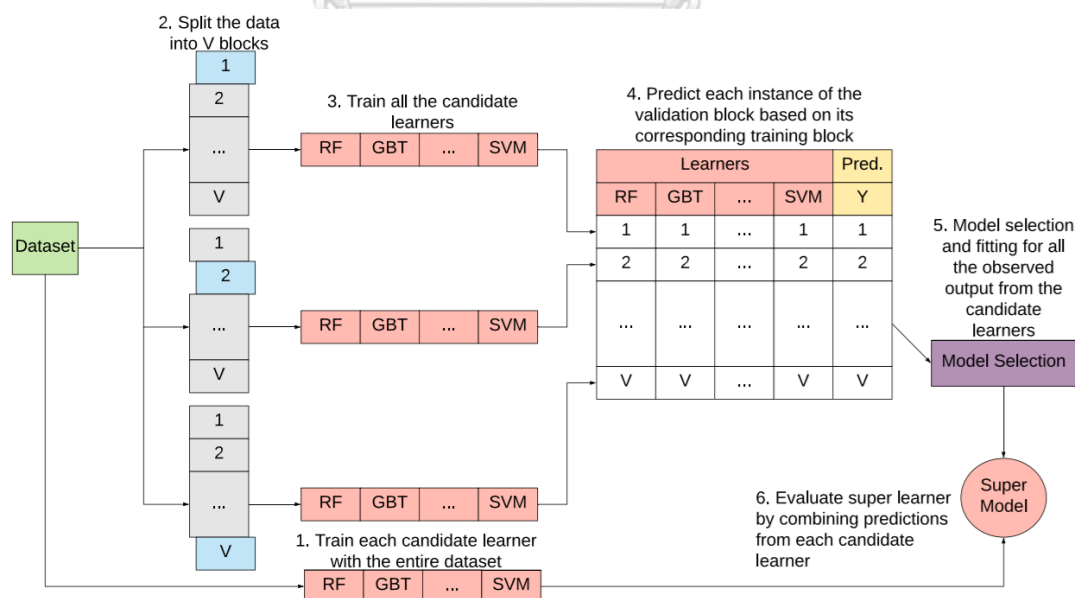


Figure 7 Example structure of Super Learner algorithm (Neto et al., 2020)

### 2.2.3 Classification Metrics

The evaluation of binary classification problems is basically based on the confusion matrix as shown in Table 5.

**Table 5** Confusion matrix

	Actual Positive Class	Actual Negative Class
Predicted Positive Class	True positive	False Positive
Predicted Negative Class	False Negative	True Negative

#### Accuracy

Accuracy is a widely used metric for classification problems, defined as the ratio of correctly predicted observations over total observations.

$$\text{Accuracy} = \frac{\text{True Positive} + \text{True Negative}}{\text{Total observations}} \quad (26)$$

Accuracy is easy to use and understand, but with imbalanced datasets, the result can be misleading due to less favor towards minority class (Chawla et al., 2004).

#### Precision

Precision is used to measure the ratio of actual positive observations among predicted positive observations.

$$\text{Precision} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Positive}} \quad (27)$$

#### Recall

Recall or Sensitivity is used to measure the ratio of predicted positive observations among actual positive observations.

$$\text{Recall} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Negative}} \quad (28)$$

#### F-Measure or F1 score

F-Measure or F1 score is the harmonic mean between Precision and Recall, it combines Precision and Recall into one metric. For imbalanced data, maximizing Recall often decreases Precision, because increasing True Positive often increases False Positive. F1 score is widely used

as a performance metric for imbalanced data, as it expresses both Precision and Recall, which refers to the overall predictive performance of the positive class.

$$F1 \text{ score} = \frac{2 * Precision * Recall}{Precision + Recall} \quad (29)$$

### Specificity

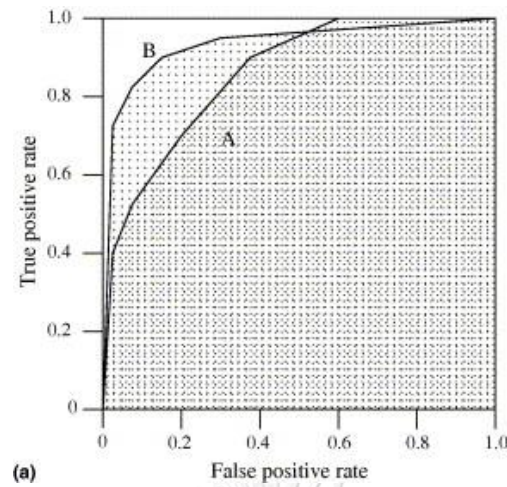
Specificity is used to measure the ratio of predicted negative observations among actual negative observations, just like the Recall of negative class. The inverse of Specificity, known as False Positive rate (1 – Specificity), is calculated in plotting ROC curve (Receiver operating characteristic curve).

$$Specificity = \frac{True \ Negative}{True \ Negative + False \ Positive} \quad (30)$$

All previously mentioned metrics are single-threshold metrics, which represent only for individual thresholds of a model and cannot measure the overall performance across various thresholds.

### Receiver operating characteristic curve (ROC curve) and Area under curve (AUC)

Receiver operating characteristic curve (ROC curve) is a plot showing the trade-off between Sensitivity and Specificity across thresholds. Area under curve (AUC) is an area under ROC curve, showing the probability that the model will value a random positive observation over a random negative observation. AUC is also used as a metric for evaluating model's performance, the model is better when AUC gets close to 1. Figure 8 shows an example of ROC curve from 2 models, A and B, where AUC is the area shaded. The inverse of Specificity, known as False Positive rate (1 – Specificity), is plotted on the x-axis, while the Sensitivity or Recall, known as True Positive rate, is plotted on the y-axis. As the True Positive rate increases, the False Positive rate increases, which shows the trade-off between Sensitivity and Specificity. Model B has better performance than Model A since it has a higher AUC (Fawcett, 2006).



**Figure 8** Example of ROC curve and AUC (Fawcett, 2006)

### Precision-Recall curve

Precision-Recall curve is a plot showing the trade-off between Precision and Recall (Sensitivity). The curve is plot across various thresholds, similar to ROC curve. It can measure the overall performance of models and is useful for model comparison (Saito & Rehmsmeier, 2015).

For imbalanced data, Precision-Recall curve is more informative than ROC curve because ROC curve only demonstrates value calculated from columns in the confusion matrix (Sensitivity and Specificity). Thus, ROC curve will not change if the proportion of positive to negative observations changes and may provide an overly optimistic result for large skewed data. On the other hand, Precision-Recall curve uses value calculated from both rows and columns in the confusion matrix, so it can provide a more informative evaluation. Figure 9 shows ROC curve and Precision-Recall curve of 2 models. When the ratio of positive class to negative class changes from 1:1 to 1:10, the Precision-Recall curve shows worse performance while the ROC curve does not change, which can lead to an overly optimistic result (Davis & Goadrich, 2006; Fawcett, 2006; Saito & Rehmsmeier, 2015).

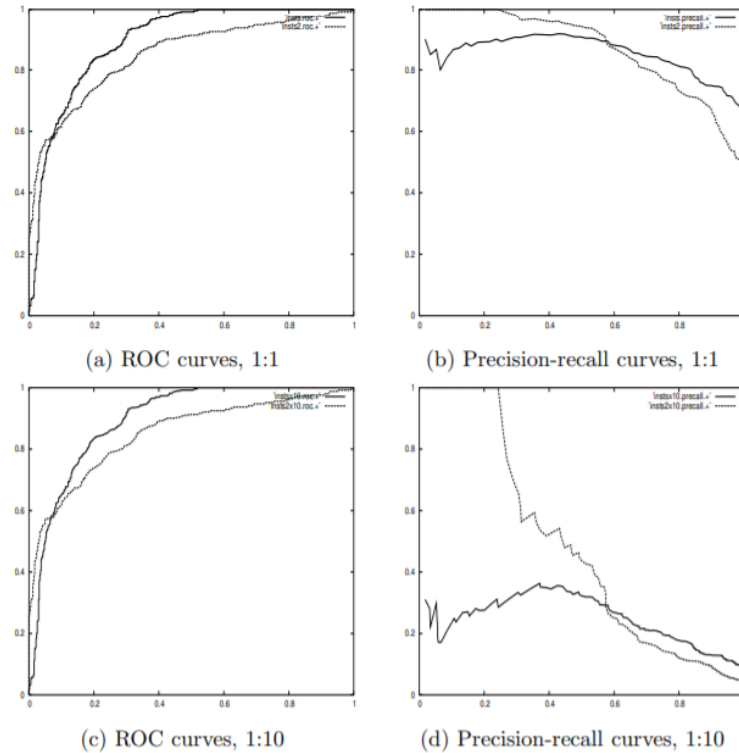


Figure 5. ROC and precision-recall curves under class skew.

Figure 9 ROC and Precision-Recall Curve under skewed data (Fawcett, 2006)

## 2.2.4 SHAP (Shapley Additive eXplanation) value

Despite the understandable structure of tree-based models, the effect of significant features in the model needs to be interpreted. It is straightforward to obtain 'Feature Importance' through the trained model, which is calculated from decreasing in Gini impurity or Entropy and the possibility of reaching that feature's nodes.

Unfortunately, the importance ranking directly generated from the models is not enough to explain features' contribution to the outcome. To clarify, though it showed which features are important, how the values of each feature affect the outcome is unknown. Thus, the additive feature attribution methods are used in this study as shown in Equation 31.

$$g(z') = \phi_0 + \sum_{i=1}^M \phi_i z'_i \quad (31)$$

Where  $g$  is a linear function of the feature attribution values,  $\phi_i$  is the feature attribution value of feature  $i$ ,  $M$  is the number of features and  $z'_i \in \{0,1\}$  stands for whether  $i$  is the observed feature (Meng et al., 2021).

The value of  $\Phi$  in Equation 15 is calculated from the method based on game theory. Classic Shapley values attribute  $\Phi_i$  for each feature  $i$  can be calculated as shown in Equation 32.

$$\Phi_i = \sum_{S \subseteq F \setminus \{i\}} \frac{|S|!(|F|-|S|-1)!}{|F|!} [f_{S \cup \{i\}}(x_{S \cup \{i\}}) - f_S(x_S)] \quad (32)$$

Where  $F$  is the set of all features,  $S$  is the subset of  $F$ ,  $f_S$  is the model trained with features in set  $S$  and  $x_S$  is the dataset contained features in set  $S$ .

With some limitations in the calculation of Equation 32, a tree SHAP value estimation algorithm is implemented for using with tree-based models (Lundberg et al., 2018).



## Chapter 3 : Methodology

### 3.1 Problem Approach

As mentioned earlier, this study aims to investigate the driven factors for stops which resulted in a conviction, as well as the driven factors for weapons usage during stops. Thus, predictive models were created to find significant features for both issues. In addition, to analyze the appropriateness of an officer' decision on weapon usage, those driven factors for both issues were compared due to the assumption that an officer should decide to use weapon based on a sign of guilty suspect.

### 3.2 Data source

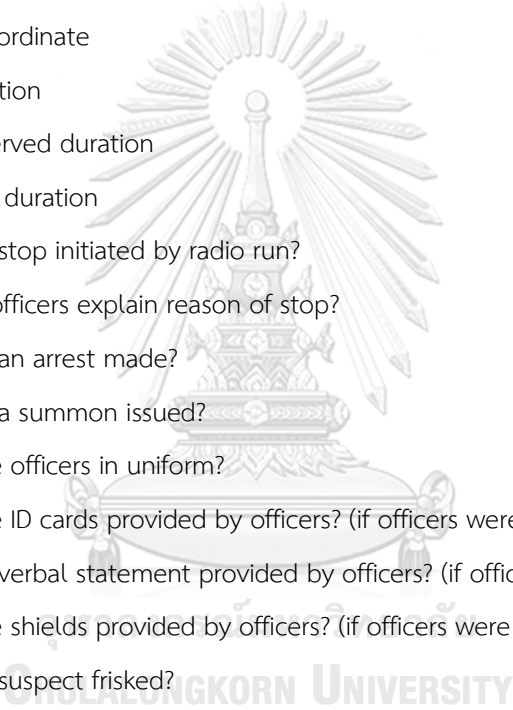
Stop, Question, and Frisk datasets used in this study were originally retrieved from the New York Police Department (NYPD) website. The data were recorded by the NYPD officers, each row in datasets represents each stop. Since the files posted on the website were annually published, the format of the SQF records may be changed through the years. Focused on the records during 2014-2019, there were major changes in dataset format after 2016. In other words, there were 2 versions of SQF records (2014-2016 and 2017-2019). In the former version (2014-2016), there were 80,753 records with 112 features and in the later version (2017-2019), there were 36,095 records with 83 features.

### 3.3 Data Preparation and Output labeling

Since there were differences between 2 versions, such as some features appeared only in one version, combining both versions was an essential process aside from the conventional data cleaning process.

To maintain consistency of the aggregated dataset, some features which did not include in both versions were dropped, some similar features were grouped in order to match those in another version, and values in some columns were grouped or changed. The features in the aggregated dataset adopted from the original versions had 50 columns in total and divided into 8 categories.

1. Date and Time
  - Year of stop
  - Month of stop
  - Date of stop

- Days in week of stop
  - Time of stop
  - 2. Authority
    - Was stop inside or outside?
    - Jurisdiction
  - 3. Location
    - Precinct
    - Borough
    - X Coordinate
    - Y Coordinate
  - 4. Police action
    - Observed duration
    - Stop duration
    - Was stop initiated by radio run?
    - Did officers explain reason of stop?
    - Was an arrest made?
    - Was a summon issued?
    - Were officers in uniform?
    - Were ID cards provided by officers? (if officers were not in uniform)
    - Was verbal statement provided by officers? (if officers were not in uniform)
    - Were shields provided by officers? (if officers were not in uniform)
    - Was suspect frisked?
    - Was suspect searched?
  - 5. Physical force used
    - Was physical force without weapon used?
    - Was physical force with gunfire used?
    - Was physical force with other weapons used?
    - Was other physical force used?
  - 6. Crime details
    - Crime suspected
    - Were other persons stopped?
    - Was any weapon found on suspect?
- 



- Was contraband found on suspect?
- Was gun found on suspect?
- Was knife found on suspect?
- Was other weapons found on suspect?

#### 7. Situational Characteristics

- Did stop relate to suspect carrying suspicious object?
- Did stop relate to suspicious appearance of the suspect?
- Did stop relate to suspect casing a victim or location?
- Did stop relate to suspect acting as a lookout?
- Did stop relate to drug transaction?
- Did stop relate to identified crime pattern?
- Did stop relate to proximity to crime scene?
- Did stop relate to evasive or other actions? (Combined due to discontinuous of features. Including furtive movements, evasive response to questioning, refuse to comply with officers' directions, change direction at sight of officers, verbal threats by suspect, and others)

#### 8. Suspect Characteristics

- Suspect's sex
- Suspect's race
- Suspect's age
- Suspect's height
- Suspect's weight
- Suspect's hair color
- Suspect's eye color
- Suspect's build

Outcome columns were calculated from related features. A suspect was "Guilty" if an arrest or issuing a summon occurred in stop records. Police's physical force usage was divided into 3 levels. The definition in each outcome column is described in Table 6.

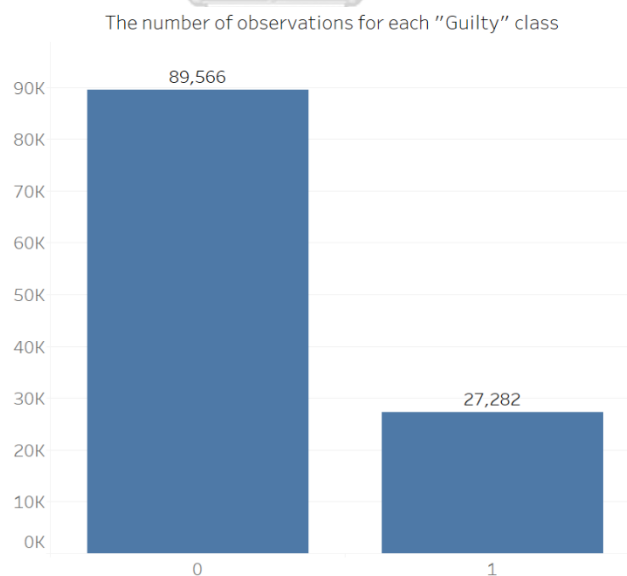
**Table 6** Outcome description

Outcome label	Description
Guilty	1 if an arrest made or summon issued 0 otherwise
Level of force	0 if no physical force or physical force without weapon 1 if physical force with other weapons, without gunfire 2 if physical force with gunfire 3 if only other physical force

Finally, duplicate records had been removed. The combined dataset totally had 116,574 records, with 52 columns.

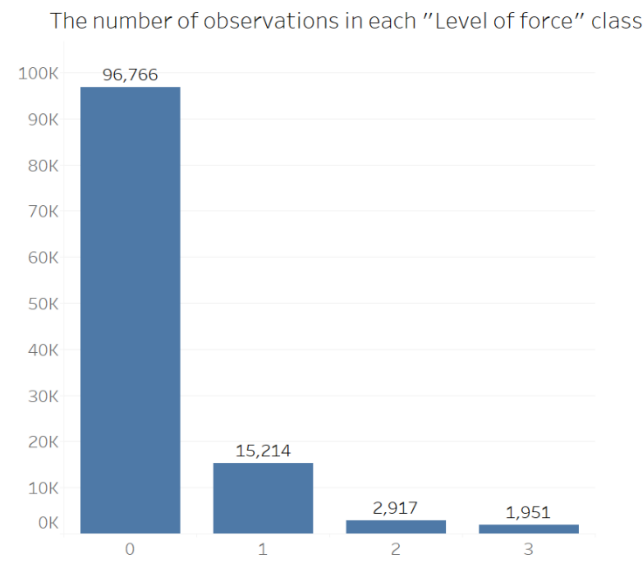
### 3.4 Exploratory Data Analysis

After combining datasets and labeling the outcome columns, the dataset was explored. As shown in Figure 10, out of 116,574 observations, 23.40% fell into the “Guilty=1” class.

**Figure 10** The number of observations for each “Guilty” class

For level of force column, out of 116,574 observations, 83.01% (96,766 observations) fell into the “Level of force = 0” class, 13.05% (15,214 observations) fell into the “Level of force = 1” class, 2.50% (2,917 observations) fell into the “Level of force = 2” class and 1.67% (1,951 observations) fell into the “Level of force = 3” class, as shown in Figure 11. Since the “Level of

force = 3” class is defined as only other physical force used, which is ambiguous, the class was excluded from the analysis process.



**Figure 11** The number of observations in each “Level of force” class

### 3.5 Tools and Model Construction

This study used Python 3.6 as a programming language, operating on Jupyter Notebook, a web-based application for data analysis. Python libraries, such as Pandas and NumPy, were implemented as tools for data preparation. For data visualization, Tableau 2020.3 along with Python libraries such as Matplotlib and Seaborn running on Jupyter Notebook were used. For training and optimizing models as well as evaluation, Python libraries such as XGBoost and Scikit-learn were implemented.

The steps of model training are as follows:

#### 1. Data adjustment

1.1 Drop some columns which directly correlated to the outcome of the predictions or were not causation of the outcome. Those included “Was an arrest made?”, “Was a summon issued?”, “Was suspect frisked?”, “Was suspect searched?”, “Stop duration”, “Did officers explain the reason for stop?”, “Were ID cards provided by officers?”, “Was verbal statement provided by officers?”, “Were shields provided by officers?”, “Was contraband found on suspect?” and all “Physical forced used” columns. As a result, the dataset had 36 features in total.

1.2 Transform categorical features into numeric values for training models, the transformation can be conducted in various ways, such as ordinal encoding and one-hot encoding. Ordinal encoding was used to transform binary features or categorical features of which the values orderly related to one another, such as days of a week (Monday to Sunday), otherwise, one-hot encoding was used, such as crime suspected.

1.3 Some continuous numerical features were transformed into discrete values by binning, such as height of suspects.

## 2. Dividing dataset

Model training was performed using the prepared dataset, which was divided into 3 sets: Training set (60%), Validation set (20%) and Test set (20%) Due to the imbalance of the dataset, the sampling was stratified by using a stratified train-test split in Scikit-learn.

## 3. Model training and optimization

Training set was used for fitting the model in the training period. The classifier of each technique has various parameters, tuning the parameters was carried out using GridSearchCV in Scikit-learn to search over parameter values for the best performance. The steps of parameters tuning for each model are as follows:

3.1 Define specific ranges of each parameter to search over for the best value

3.2 Divide Training set into 3 subsets, for a set of parameters, do the cross-validation by training 2 subsets using those parameters and evaluate the performance with another subset. Repeat the steps for 2 times by using different training and evaluating subsets.

3.3 Repeat step 3.2 with all possible combination of the parameters.

3.4 Average the F1 scores of each combination of parameters from cross-validation process.

3.5 Choose the combination of parameters that had the best performance.

For Super Learner, the meta learners were optimized as well.

## 4. Dropping insignificant features

4.1 Train the tuned model with all features using Training set.

4.2 Obtain the importance of features and sort from minimum to maximum value.

4.3 Drop the least significant feature. Train the model with the remaining features and evaluate the model with the validation set.

4.4 Repeat step 4.3 until F1 score does not improve.

4.5 Use the remaining features that give the best performance.

5. Repeat step 3 with the remaining features

6. Threshold Adjustment

Threshold is the cut-off probability to determine whether each sample is in the class. Basically, the models predict the probability of being in the outcome class for each record in the test set. For binary classification, the default threshold is set to 0.5, which is, if the predicted probability for the positive class of a sample is more than 0.5, then the sample will result in the positive class. For imbalance data, adjusting threshold plays a significant role in optimizing the evaluation metrics, as mentioned in Chapter 2.

7. Evaluate the result of each model

7.1 Combine Training and Validation set and divide into 4 subsets

7.2 Do 4-folds cross-validation by training 3 subsets with the optimal model and evaluating with another subset.

7.3 Average the performance from all folds.

8. Compare the results and choose the best model

9. Calculate SHAP value for each feature from the best model

10. Test the best model with Test set to obtain the final result.

### 3.6 Evaluation metric

For Guilty Prediction, which is binary classification with imbalance, Area Under Curve of Precision-Recall Curve was used for comparing the model's performance among techniques, as well as selecting the best threshold for each model which was calculated from the threshold that optimizes the F1 score.

For Level of Force Prediction, which is multiclass classification with imbalance, F1 score of minority class (level 1 and level 2) was used for comparing the model's performance among techniques to harmonize between Precision and Recall. So, the macro average (simple average with no weight considered) of F1 score was implemented.

F1 score, Accuracy, Precision and Recall were measured for evaluating the best model's performance when testing with Test set.

Finally, for investigating the contribution of features in all techniques, SHAP values for each feature were calculated from the best models of each technique.

### 3.7 Super Learner experiment

To explore various structural designs for Super Learner, experiments were divided into 2 sections, Guilty Prediction (section A) and Level of Force Prediction (section B). Each section contains 4 experiments as described in Table 7. Each section was divided into 4 experiments based on base models; (1) Tree-based without tuning, (2) Tree-based tuned, (3) Tree-based, Logistic Regression, and Gaussian Naïve Bayes, all without tuning, and (4) Tree-based, Logistic Regression, and Gaussian Naïve Bayes, all were tuned.

**Table 7** Super Learner experimental plan

Prediction	Experiment	Base models	Meta model
Guilty Prediction	A1	DF, RF, XGB (No tuning)	DT, RF, XGB, LR, MLP (5 meta models for each experiment)
	A2	DF, RF, XGB (tuned)	
	A3	DF, RF, XGB, LR, GB (No tuning)	
	A4	DF, RF, XGB, LR, GB (tuned)	
Level of Force Prediction	B1	DF, RF, XGB (No tuning)	DT, RF, XGB, LR, MLP (5 meta models for each experiment)
	B2	DF, RF, XGB (tuned)	
	B3	DF, RF, XGB, LR, GB (No tuning)	
	B4	DF, RF, XGB, LR, GB (tuned)	

\*DF = Decision Tree, RF = Random Forest, XGB = XGBoost, LR = Logistic Regression, GB = Gaussian Naïve Bayes,

MLP = Multilayer Perceptron Neural Network

\* tuned = single model optimization

## Chapter 4 : Results and Discussion

### 4.1 Results of tree-based models

#### 4.1.1 Guilty Prediction

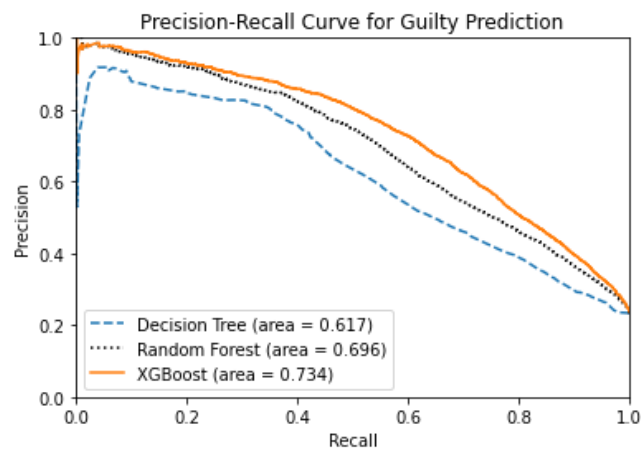
The models of all techniques were optimized by parameters and threshold tuning. The ranges and optimum values for each technique are demonstrated in Appendix.

After tuning the parameters and threshold, cross-validation was conducted. The average results from 4-fold cross-validation of all techniques are shown in Table 8.

**Table 8** Average results from 4-fold cross-validation for Guilty Prediction

Models	Metrics (4 fold cross validation)				
	AUC of PCR	Accuracy	F1 Score	Precision	Recall
Decision Tree	0.612	0.801	0.569	0.578	0.560
Random Forest	0.692	0.824	0.622	0.626	0.618
XGBoost	<b>0.731</b>	<b>0.841</b>	<b>0.662</b>	<b>0.657</b>	<b>0.667</b>

XGBoost had the best performance among all models since it obtained the highest score in all metrics, followed by Random Forest and Decision Tree, which had the lowest score in all metrics. All techniques reached good performance with more than 80% accuracy especially XGBoost, which obtained 84.1% accuracy while maintained both good Precision and good Recall at 65.7% and 66.7% respectively. Precision-Recall Curves and AUC from the best models of each technique are shown in Figure 12, XGBoost also outperformed other models across all thresholds with the highest AUC.



**Figure 12** Precision-Recall Curves for Guilty Prediction

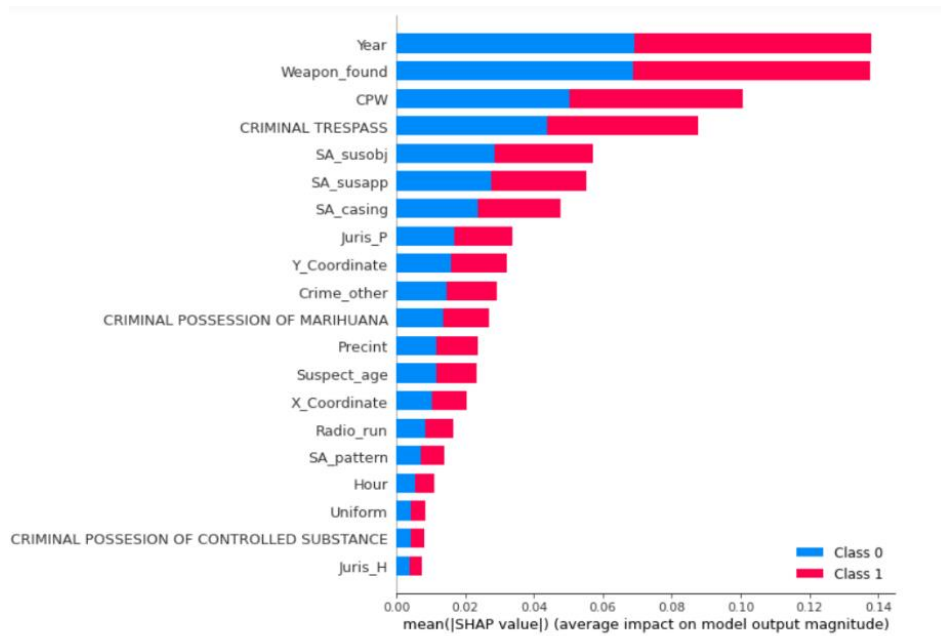
The best XGBoost model was evaluated using Test set and the results are shown in Table 9. The average results from 4-fold cross-validation mentioned above are also shown in this table for comparison. It showed that the model was reliable because the results of predicting the unseen dataset (Test set) conformed to those of the training period.

**Table 9** Performance of the best XGBoost model on Test set for Guilty Prediction (comparing with 4-fold cross-validation on Training set)

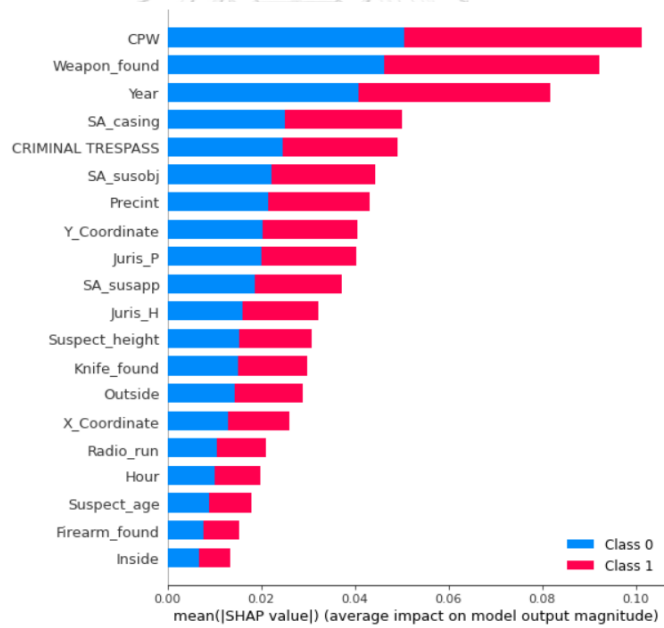
Results	Metrics (4 fold cross validation)			
	Accuracy	F1 Score	Precision	Recall
Test set	0.840	0.659	0.659	0.659
4-fold cross-validation	0.841	0.662	0.657	0.667

In terms of important factors, the best models from each technique were analyzed and SHAP values of all factors were calculated. The important features ranked by the average magnitude of SHAP values are illustrated in Figure 13 – 15. The top 10 features are described in Table 10.





**Figure 13** Important features in the best Decision Tree model for Guilty Prediction based on average magnitude of SHAP values (class 0 = Not Guilty, class 1 = Guilty)



**Figure 14** Important features in the best Random Forest model for Guilty Prediction based on average magnitude of SHAP values (class 0 = Not Guilty, class 1 = Guilty)

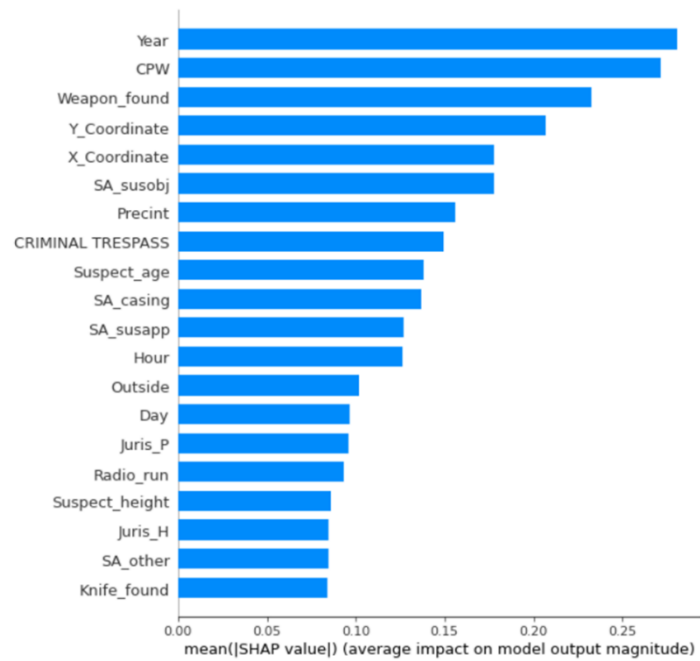
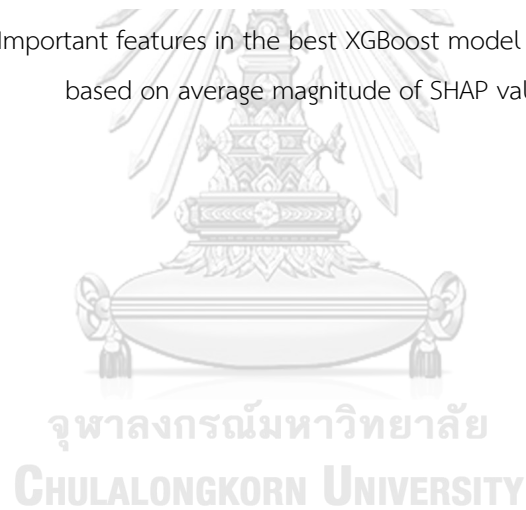


Figure 15 Important features in the best XGBoost model for Guilty Prediction based on average magnitude of SHAP values

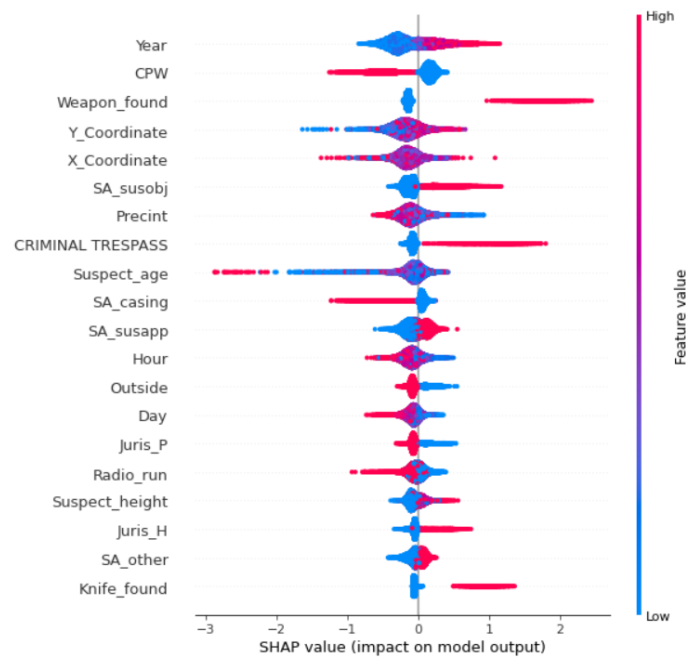


**Table 10** Top 10 important features for Guilty Prediction

Decision Tree	Random Forest	XGBoost
<b>1. Year of the stop</b>	<b>1. Was crime suspected Criminal Possession of Weapon?</b>	<b>1. Year of the stop</b>
<b>2. Was any weapon found on suspect?</b>	<b>2. Was any weapon found on suspect?</b>	<b>2. Was crime suspected Criminal Possession of Weapon?</b>
<b>3. Was crime suspected Criminal Possession of Weapon?</b>	<b>3. Year of the stop</b>	<b>3. Was any weapon found on suspect?</b>
<b>4. Was crime suspected Criminal Trespass?</b>	<b>4. Did stop relate to suspect casing a victim or location?</b>	<b>4. Y coordinate of the stop</b>
<b>5. Did stop relate to suspect carrying suspicious object?</b>	<b>5. Was crime suspected Criminal Trespass?</b>	<b>5. X coordinate of the stop</b>
<b>6. Did stop relate to suspicious appearance of the suspect?</b>	<b>6. Did stop relate to suspect carrying suspicious object?</b>	<b>6. Did stop relate to suspect carrying suspicious object?</b>
<b>7. Did stop relate to suspect casing a victim or location?</b>	<b>7. Precinct (New York City regions)</b>	<b>7. Precinct (New York City regions)</b>
<b>8. Did Patrol Service Bureau have jurisdiction?</b>	<b>8. Y coordinate of the stop</b>	<b>8. Was crime suspected Criminal Trespass?</b>
<b>9. Y coordinate of the stop</b>	<b>9. Did Patrol Service Bureau have jurisdiction?</b>	<b>9. Suspect's age</b>
<b>10. Was crime suspected other crime?</b>	<b>10. Did stop relate to suspicious appearance of the suspect?</b>	<b>10. Did stop relate to suspect casing a victim or location?</b>

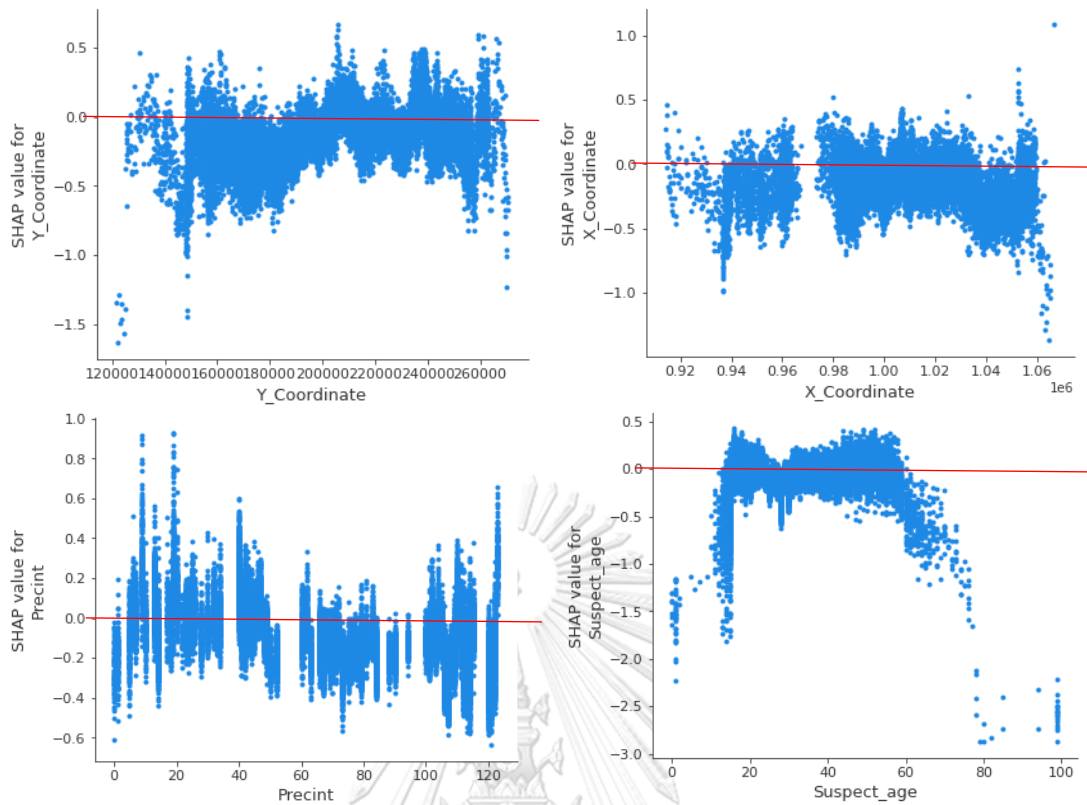
According to Table 10, the bold features are ones that appear in all columns, which means they were important in all techniques. The important features were similar among techniques as most of the features were bold. Moreover, all techniques obtained the same top 3 important features which were “Year of stop”, “Was any weapon found on suspect?” and “Was crime suspected Criminal Possession of Weapon?”.

Since the XGBoost model had the best performance among others, its effects of features on outcome column were focused. Raw SHAP values calculated from samples are plotted in Figure 16 to show effects toward Guilty class from features.



**Figure 16** Effects of important features toward Guilty class based on SHAP value

From Figure 16, among top 10 features, “Y coordinate”, “X coordinate”, “Precinct” and “Suspect’s age” did not clearly show direction of values that led to Guilty class. Thus, SHAP value for those features were plotted as illustrated in Figure 17



**Figure 17** SHAP value of “Y coordinate”, “X coordinate”, “Precinct” and “Suspect’s age”

According to Figure 17, entire range of “Y coordinate”, “X coordinate” and “Precinct” could possibly lead to Guilty class as there were positive SHAP value (plotted above the red line) over entire range. However, the range of “Suspect’s age” that had positive SHAP value (plotted above the red line) was in the range of around 15– 60 years old. In summary, effects of Top 10 important features on the outcome column are shown in Table 11.

**Table 11** Effects toward Guilty class of important features the best XGBoost model  
(Features which were important in all techniques were also bold)

Top 10 Important Features	Value that led to Guilty
<b>1. Year of the stop</b>	Close to 2019
<b>2. Was crime suspected Criminal Possession of Weapon?</b>	No
<b>3. Was any weapon found on suspect?</b>	Yes
<b>4. Y coordinate of the stop</b>	-
5. X coordinate of the stop	-
<b>6. Did stop relate to suspect carrying suspicious object?</b>	Yes
7. Precinct (New York City regions)	-
<b>8. Was crime suspected Criminal Trespass?</b>	Yes
9. Suspect's age	Around 15 - 60
<b>10. Did stop relate to suspect casing a victim or location?</b>	Yes

#### 4.1.2 Level of Force Prediction

The models of all techniques were optimized by parameters tuning. The ranges and optimum values for each technique are demonstrated in Appendix.

After tuning the parameters and threshold, cross-validation was conducted. The average results from 4-fold cross-validation of all techniques are shown in Table 12.

**Table 12** Average results from 4-fold cross-validation for Level of Force Prediction

Models	Accuracy	F1 score			Precision		Recall	
		Level 1	Level 2	Macro average	Level 1	Level 2	Level 1	Level 2
Decision Tree	0.694	0.311	0.187	0.249	0.245	0.135	0.426	0.306
Random Forest	0.788	0.382	0.289	0.336	0.351	0.273	0.420	0.309
XGBoost	<b>0.806</b>	<b>0.401</b>	<b>0.366</b>	<b>0.384</b>	<b>0.376</b>	<b>0.389</b>	<b>0.430</b>	<b>0.346</b>

XGBoost had the best performance among all models since it obtained the highest score in all metrics, followed by Random Forest and Decision Tree, which had the lowest score in all metrics except Recall of Level 1. XGBoost reached good performance with more than 80% accuracy while F1 scores of the minority classes, level 1 and level2, were 40.1% and 36.6% respectively.

The best XGBoost model was evaluated using Test set and the results are shown in Table 13. The average results from the 4-fold cross-validation mentioned above are also shown in this table for comparison. It showed that the model was reliable because the results of predicting the unseen dataset (Test set) conformed to those of the training period.

**Table 13** Performance of the best XGBoost model on Test set for Level of Force Prediction (comparing with 4-fold cross-validation on Training set)

Results	Accuracy	F1 score			Precision		Recall	
		Level 1	Level 2	Macro average	Level 1	Level 2	Level 1	Level 2
Test set	0.804	0.407	0.350	0.379	0.375	0.373	0.444	0.330
4 folds cross-validation	0.806	0.401	0.366	0.384	0.376	0.389	0.430	0.346

In terms of important factors, the best models from each technique were analyzed and SHAP values of all factors were calculated. The important features ranked by the average magnitude of SHAP values are illustrated in Figure 18 – 20, and the top 10 features are described in Table 14.

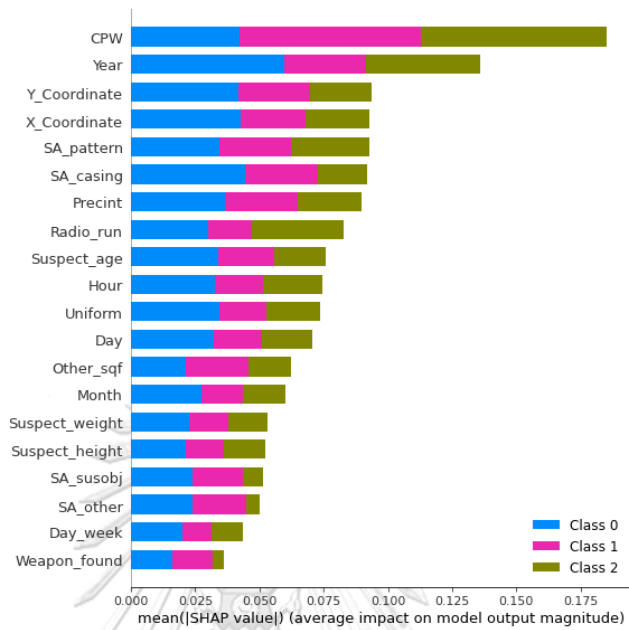


Figure 18 Important features in the best Decision Tree model for Level of Force Prediction based on average magnitude of SHAP values (Class = Level)

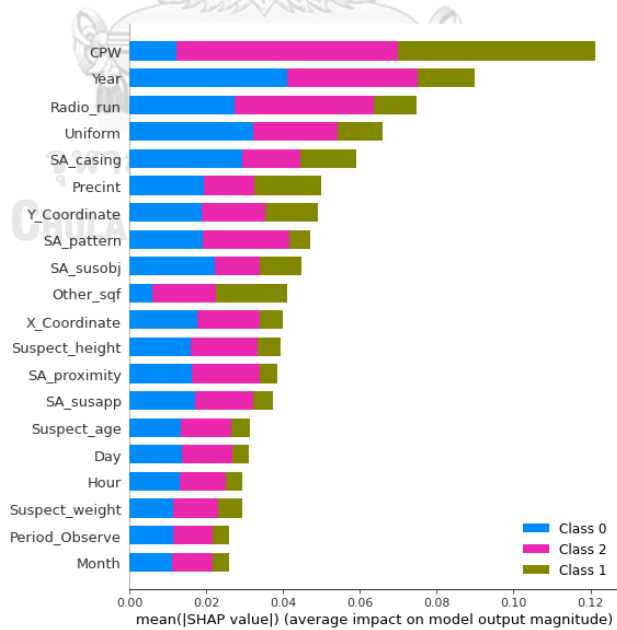
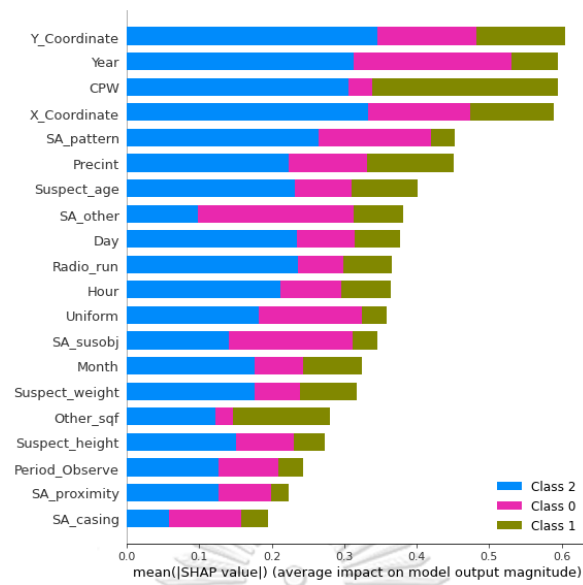


Figure 19 Important features in the best Random Forest model for Level of Force Prediction based on average magnitude of SHAP values (Class = Level)





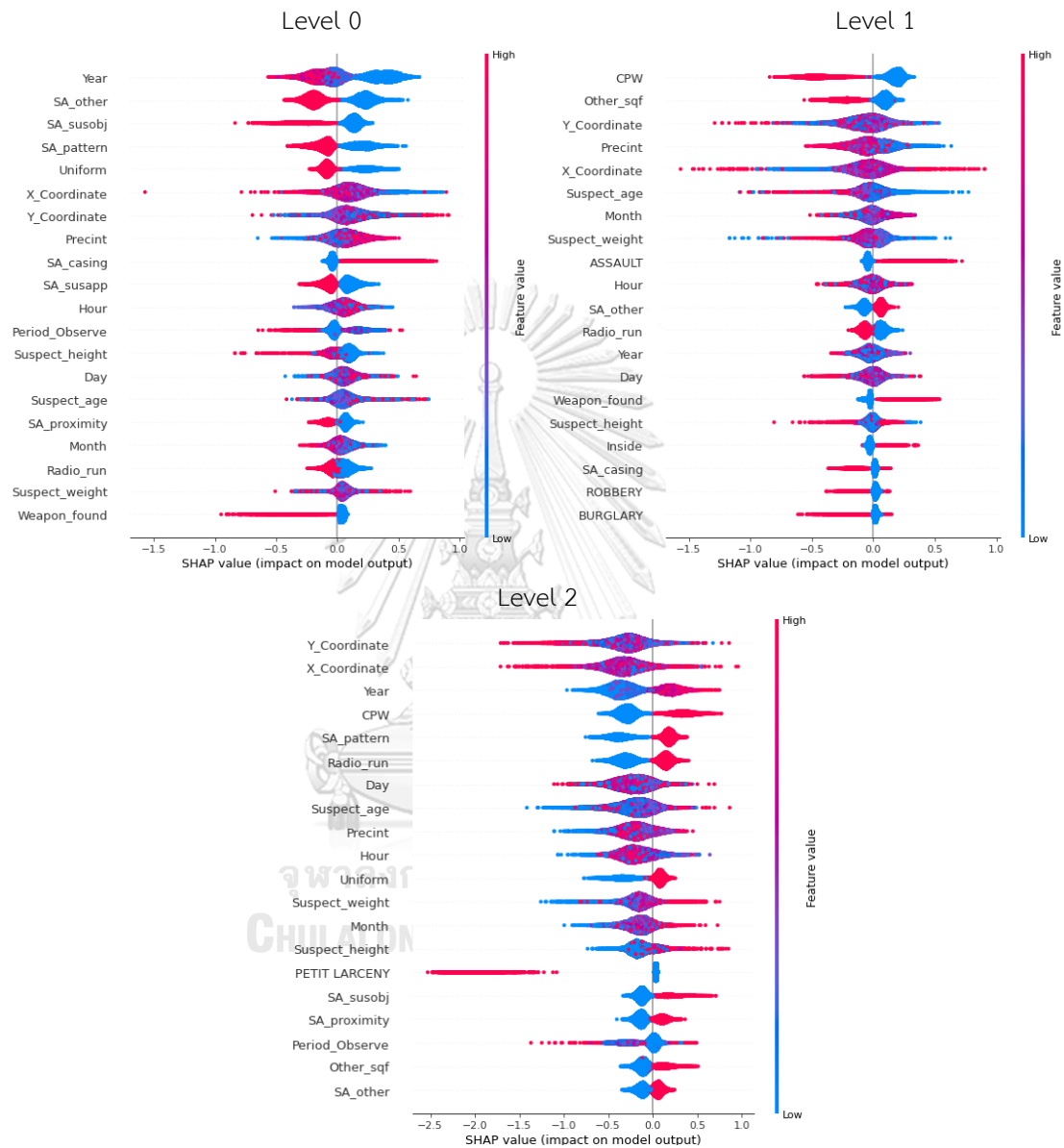
**Figure 20** Important features in the best XGBoost model for Level of Force Prediction based on average magnitude of SHAP values (Class = Level)

**Table 14** Top 10 important features for Level of Force Prediction

Decision Tree	Random Forest	XGBoost
<b>1. Was crime suspected Criminal Possession of Weapon?</b>	<b>1.. Was crime suspected Criminal Possession of Weapon?</b>	<b>1. Y Coordinate of the stop</b>
<b>2. Year of the stop</b>	<b>2. Year of the stop</b>	<b>2. Year of the stop</b>
<b>3. Y Coordinate of the stop</b>	<b>3. Was stop initiated by radio run?</b>	<b>3. Was crime suspected Criminal Possession of Weapon?</b>
<b>4. X Coordinate of the stop</b>	<b>4. Were officers in uniform?</b>	<b>4. X Coordinate of the stop</b>
<b>5. Did stop relate to identified crime pattern?</b>	<b>5. Did stop relate to suspect casing a victim or location?</b>	<b>5. Did stop relate to identified crime pattern?</b>
<b>6. Did stop relate to suspect casing a victim or location?</b>	<b>6. Precinct (New York City regions)</b>	<b>6. Precinct (New York City regions)</b>
<b>7. Precinct (New York City regions)</b>	<b>7. Y Coordinate of the stop</b>	<b>7. Suspect's age</b>
<b>8. Was stop initiated by radio run?</b>	<b>8. Did stop relate to identified crime pattern?</b>	<b>8. Did stop relate to evasive or other actions?</b>
<b>9. Suspect's age</b>	<b>9. Did stop relate to suspect carrying suspicious object?</b>	<b>9. Day of the stop</b>
<b>10. Hour of the stop</b>	<b>10. Were other persons stopped?</b>	<b>10. Was stop initiated by radio run?</b>

According to Table 14, same as Guilty Prediction, the bold features are ones that appear in all columns, which means they were important in all techniques. The important features were similar among techniques as most of the features were bold. Note that “Year of the stop” and “Was crime suspected Criminal Possession of Weapon?” were ranked in the top 3 for all techniques in both Guilty and Level of Force Prediction.

Since the XGBoost model had the best performance among others, its effects of features on the outcome column were focused. Raw SHAP values calculated from samples are plotted in Figure 21 to show effects toward each Level from features.



**Figure 21** Effects of important features toward each Level based on SHAP value

From Figure 21, ranks of features were different for each Level and different from the overall rank shown in Figure 20. This is because it was ranked by sum of the magnitude of each Level. Thus, only the top 10 features from the overall rank were focused. From the plots, some

features did not clearly show the direction of values that led to each Level. Therefore, SHAP values of those features were plotted and investigated.

As a result, some features including “Y coordinate”, “X coordinate”, “Day of the stop” and “Precinct” could lead to any Level in a wide range of values (similar to the plots shown in Figure 17). Some features clearly showed the direction of values in some Levels, for example, “Year of the stop” showed the clear direction in Level 0 and 2, but not in Level 1. The effects of Top 10 important features on the outcome column are summarized in Table 15. Additionally, the plot of SHAP values toward Level 1 for “Suspect’s age” is shown in Figure 22, in order to clarify the value in Table 15.

**Table 15** Effects toward each Level of important features in the best XGBoost model  
(Features which were important in all techniques were also bold)

Top 10 Important Features	Value that leads to each level		
	0	1	2
<b>1. Y Coordinate of the stop</b>	-		
<b>2. Year of the stop</b>	Close to 2014	-	Close to 2019
<b>3. Was crime suspected Criminal Possession of Weapon?</b>	-	No	Yes
4. X Coordinate of the stop	-		
<b>5. Did stop relate to identified crime pattern?</b>	No	-	Yes
<b>6. Precinct (New York City regions)</b>	-		
7. Suspect’s age	-	Around 15-60	-
8. Did stop relate to evasive or other actions?	No	Yes	Yes
9. Day of the stop	-		
<b>10. Was stop initiated by radio run?</b>	No	No	Yes

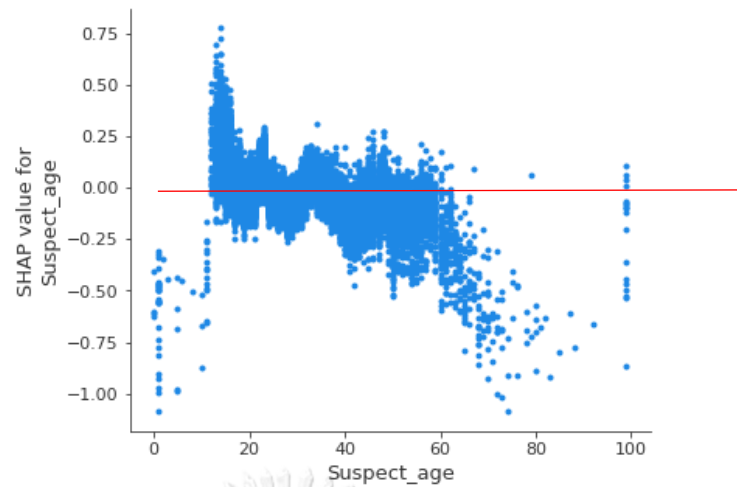


Figure 22 SHAP values toward Level 1 for “Suspect’s age”

## 4.2 Discussion of tree-based models

### 4.2.1 Results interpretation

Although accuracy is a basic metric for model evaluation, focusing only on accuracy may lead to misinterpretation when it comes to an imbalanced dataset. Regarding Guilty Prediction, if all samples were predicted as “Not Guilty”, the model would simply achieve 76.63% accuracy while all “Guilty” cases were wrongly predicted. That was why Precision and Recall of the “Guilty” class were taken into consideration in this problem. This issue was also applied to the Level of Force Prediction, in which the “Level 1” class and “Level 2” class were minority but interesting classes.

As previously mentioned in Chapter 2, maximizing Precision often decreases Recall, and vice versa. Referring to the context of this problem, low Precision means when the model predicted that a record was “Guilty” or experienced force at “Level 1 or 2”, most of the predictions were wrong. Meanwhile, low Recall means the model cannot recall most of the records which were actually “Guilty” or experienced force at “Level 1 or 2”. Since both “low Precision” and “low Recall” cases were unacceptable, the F1 score was focused to maximize both metrics simultaneously.

From the results of Test set in Chapter 4, the best XGBoost model for Guilty Prediction obtained F1 score at 65.9% with both Precision and Recall at 65.9%. This means that 65.9% of all stops that the model predicts “Guilty” are really “Guilty” (precision) and means that 65.9% of all

stops that are “Guilty” are correctly predicted (recall). This shows that the model can increase the possibility of detecting the minority class which was originally 23.37% (stops with “Guilty” are 23.37% of the population).

For Level of Force Prediction, the best XGBoost model obtained F1 score for “Level 1” at 40.7% (37.5% Precision and 44.4% Recall), and for “Level 2” at 35.0% (37.3 Precision and 33.0% Recall). While the performance seems inferior to those in Guilty Prediction, the model increased the significant probability of detecting the minority classes, which were originally 13.04% and 2.49% of the population for “Level 1” and “Level 2” respectively. Aiming to optimize F1 score of both Level 1 and Level 2, the process optimized macro average F1 score which resulted in similar F1 score for both Levels. In addition, obtaining 80.4% accuracy might be lower than 82.8% accuracy when the model just predicted “Level 0” for all samples. However, the decreased accuracy was traded off by increasing the ability to detect more severe usage of force, which was the focused issue of this project as mentioned earlier.

## 4.2.2 XGBoost Performance

According to the results, XGBoost outperformed other tree-based models, Decision Tree and Random Forest, by all metrics in both predictions. Different scheme of building trees was probably a major cause of this. Trees in Random Forest were built separately and the result came from the majority votes among the trees. Meanwhile, trees in the XGBoost model were sequentially built by the previous trees with parameters calculated to minimize loss function, then the result came from the last tree. This ability to learn from the previous trees possibly led to a better fit with the dataset.

Optimization also played a significant role in model construction. Parameter tuning and dropping insignificant features helped to increase the performance in all techniques. For example, the F1 score in XGBoost for Guilty Prediction was increased from 55.7% to 66.2% by the optimization. The optimal model contained 63 from 112 columns with tuned parameters, namely as a learning rate of 0.01 (default at 0.3), max depth of 25 (default at 6), etc.

## 4.2.3 Interpretation of important features’ effect

### 4.2.3.1 Guilty Prediction

“Year” had the largest impact on Guilty Prediction, and later years (close to 2019) led to more possibility for a record to be “Guilty”. The result conformed to Figure 2 in Chapter 1, the

plot shows the decrease in “Not Guilty” proportion, and thus the increase in “Guilty” proportion. This was a good sign that NYPD had a tendency to lessen the “Not Guilty” cases. However, as mentioned in Chapter 1, the proportion of “Not Guilty” was still high.

Crimes suspected including “Criminal Possession of Weapon” and “Criminal Trespass” were significant driving factors. From the results shown, when the police suspected a case to be Criminal Trespass, the case tended to be Guilty since trespassing was quite an explicit crime to commit. On the other hand, a case tended to be “Not Guilty” when the police suspected a case to be “Criminal Possession of Weapon”, which was an interesting finding especially because this feature had the second-largest impact on the model. Moreover, “Criminal Possession of Weapon” was the most suspected crime. All above could imply that police often assumed that suspects committed “Criminal Possession of Weapon” when they did not, and this may have contributed to high unnecessary stops.

In terms of the suspect’s actions from the police’s point of view, “Carrying suspicious object” and “Casing a victim or location” were the significant factors. Both actions led to “Guilty” cases as expected. Another important factor, the third-largest impact on the model, was “Weapon found on suspect”, which also led to “Guilty” cases. The results of this factor, together with “Carrying suspicious object”, pointed to the fact that weapons carriage was a major element of criminal in New York City. However, the effect from “Criminal Possession of Weapon” was the opposite, indicating that there was significant false suspicion in the operation.

Lastly, the result, which showed that suspects who were in the range of around 15 – 60 years old led to “Guilty” cases, was sensible. Although the location features including “X coordinate”, “Y coordinate” and “Precinct” were important, values of these features that led to “Guilty” were widespread, meaning that crime areas were spread out in all direction of NYC.

#### 4.2.3.2 Level of Force Prediction

In Level of Force Prediction, Year also played a significant role as the second-largest impact feature. The result showed that the former years (close to 2014) led to “Level 0” and the later years (close to 2019) led to “Level 2”. This corresponds to Figure 3 in Chapter 1, which indicates increases in police force usage over years.

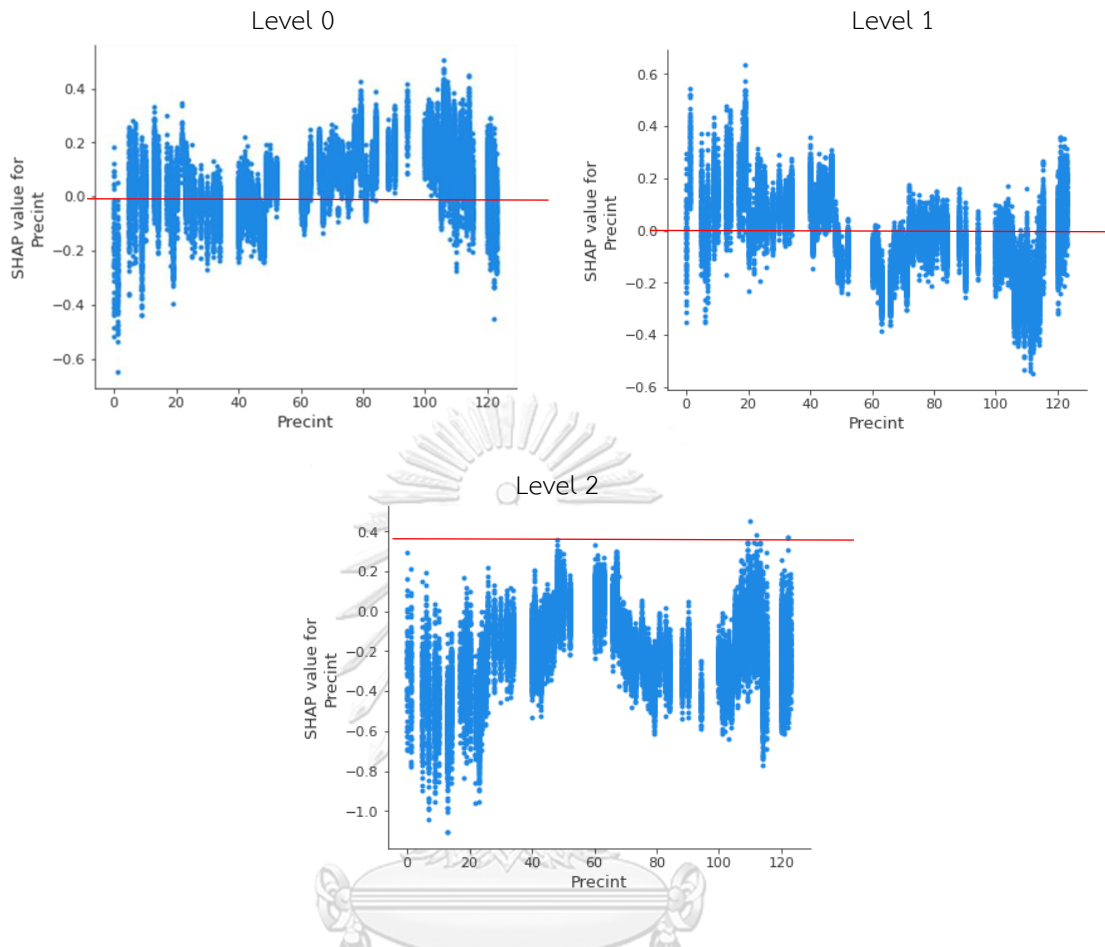
Criminal Possession of Weapon was the only significant crime suspected. According to the results, when the police suspected that a case was Criminal Possession of Weapon, they

tended to use “Level 2” of Force, and use “Level 1” when it was other crimes. This indicates the possibility of gunfire usage if the police assumed that the suspect possessed weapons. Another significant police’s action was “Stop initiated by radio run”, which led to “Level 2” if it was and led to other levels if it was not. This also suggested that cases initiated by radio run tended to experience gunfire usage. From “Street Stop Encounter Report” published on New York City website, police might receive a radio report from a third party other than personal observation. Further inspection should be done to investigate why these radio-initiated cases tended to encounter more severe force.

The suspect’s actions that had an important impact on Level of Force Prediction were “Identified Crime Pattern” and “Evasive or others”. From the results, the police possibly used “Level 2” of Force when they “Identified Crime Pattern”, and “Level 0” when they did not. This might happen because the pattern indicated violent crimes. “Evasive or others” was also a combination of multiple actions, including “Refuse to comply with officers’ directions” and “Verbal threats by the suspect”. The effect led to Level 2 when suspects doing those actions. These findings suggested that the suspect’s acts connected with violent crime and threat to officers increased the likelihood of police drawing firearms, which was also highlighted in past studies.

Finally, the results showed that suspects who were in the range around 15 – 60 years old led to “Level 1” of Force. Other important features, including “Day of the stop”, “Y Coordination”, “X Coordination” and “Precinct”, were not clearly showed the direction of values that led to specific Level. However, there were potential directions of values in “Precinct”. Plots of SHAP value toward each Level for Precinct are shown in Figure 23. From the plots, the higher Precinct number had potential leading to “Level 0”, while the lower number had potential leading to “Level 1”. The information about which Borough each Precinct number belongs to is provided in Table 16, the data was obtained from the NYPD website for better understanding.





Borough	Precinct
Manhattan	1-34
Bronx	40-52
Brooklyn	60-94
Queens	100-115
Staten Island	120-123

Figure 23 Plots of SHAP value toward each Level for Precinct

Table 16 Boroughs and Precinct numbers in New York City

#### 4.2.4 Appropriateness of the NYPD officer’s behavior in SQF practices

As discussed in the previous sections, the results of “Criminal Possession of Weapon” feature had high impact on both predictions, but there was a conflict in terms of effects interpretation. While the high number of unnecessary stops probably caused from the false

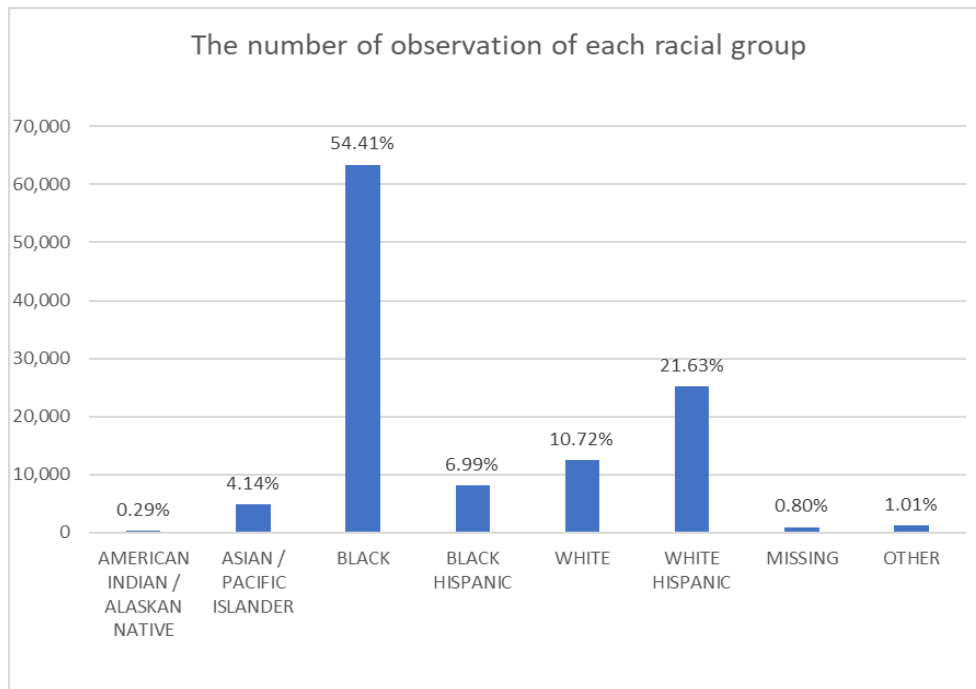
assumption of this crime suspected, its effect led to the police's gunfire usage. Moreover, the high contribution of "Weapon found" feature toward "Guilty" class pointed to the actual guilty cases with weapons carriage, but the feature did not show any importance in Level of Force Prediction for all techniques. This indicated lack of precise assumption about suspect's weapons possession during the operation, which may lead to force usage toward innocence citizen.

Suspect's actions from police's point of view appeared to be significant features in both predictions, but they were different actions for each prediction. "Carrying suspicious object" and "Casing a victim or location" led to actual guilty cases, but they were not the driving factors in force usage selection. Meanwhile, "Identified Crime Pattern" and "Evasive or others" were the driving factors for police's weapons usage, but it did not contribute to "Guilty" of the case. This conflict can be further analyzed to ensure appropriateness of police's weapons usage.

Lastly, from the findings, suspect's race was not the significant factor in any problems and techniques. Although races may not be the main driving factor from the methodology used in this project, we cannot claim that there was no different treatment in the operation among racial groups. To further investigate racial discrimination in the SQF practices, as mentioned in Chapter 1, additional hypothesis testing was done in the next section.

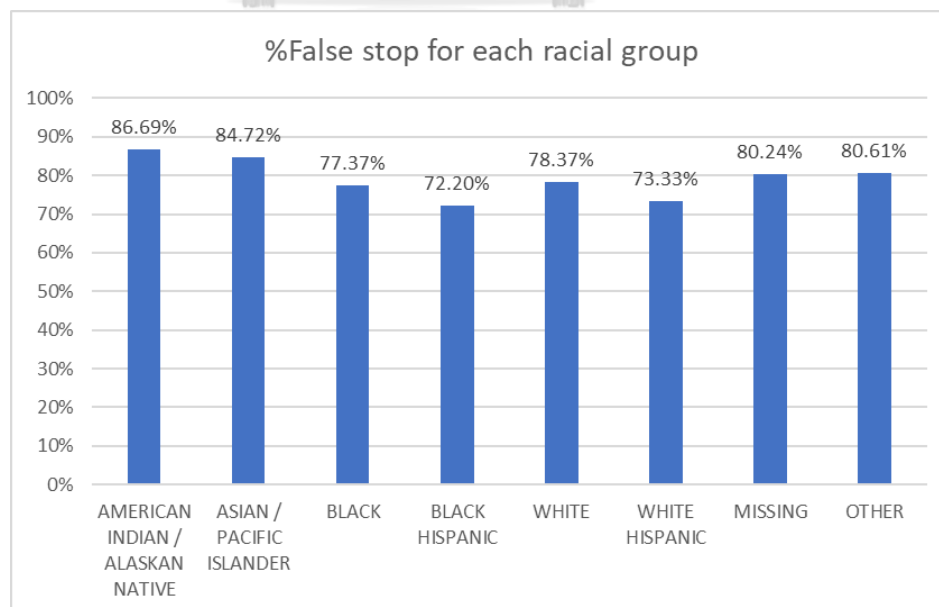
#### 4.2.5 Supplementary hypothesis testing among racial groups

There were 8 racial groups in this dataset, including "Missing" group (cases of which suspect's race were missing). As shown in Figure 24, there was the significant imbalance in those groups. More than half of all suspects stopped were "Black", and more than 80% of all suspects stopped were "Black", "White Hispanic" and "White".



**Figure 24** The number of observations for each racial group

Hypothesis testing was conducted in 2 parts, “False stop” and “Level of Force” among racial groups. A “False stop” case actually was a “Not Guilty” case because stops with no conviction were acted on innocent citizen and were unnecessary. From Figure 25, there were some differences in the proportion of “False stop” among the groups. Thus, the objective of the “False stop” part is to explore whether those differences were significant.



**Figure 25** Percentages of “False stop” for each racial group

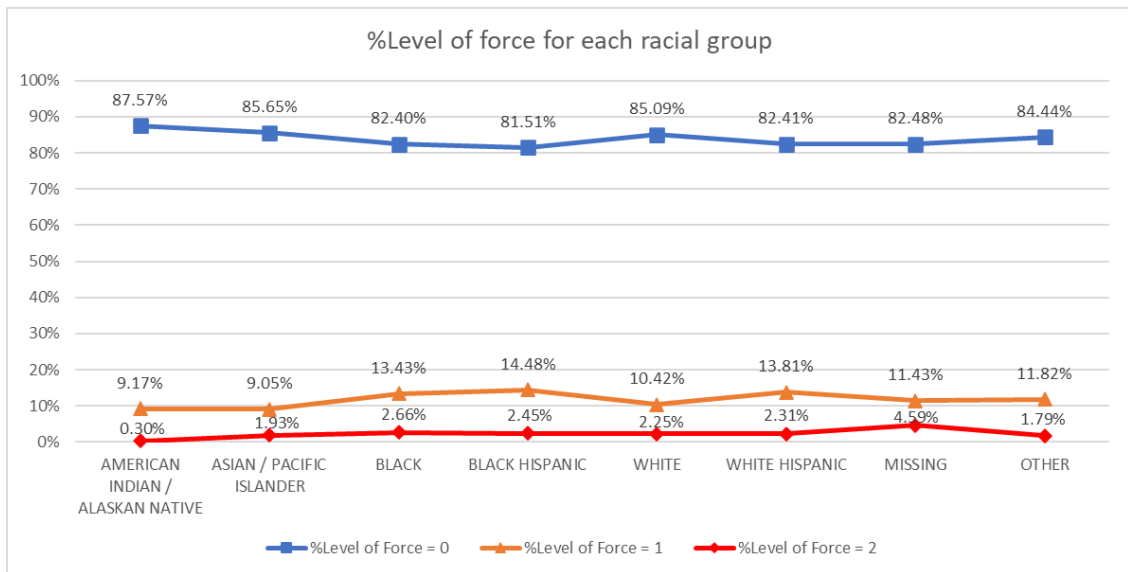
To achieve the objective of this part, Chi-square test was conducted to analyze difference in proportion of guilty among races. The results are shown in Table 17.

**Table 17** Results of hypothesis testing for “False stop” among racial groups

Variable	Value
$\chi^2$	496.2199534
No. of racial groups	8
Df	7
Critical value	14.06714045
p-value	0.00
Alpha	0.05
Result	Reject null hypothesis
Cramer's V	0.065243352
Result	Little association

The conclusion from Table 17 is that there was a significant difference in the proportion of “False stop” among racial groups at 95% confidence level. It thus suggests that the operation proceeding with some racial groups, such as “American Indian/ Alaskan Native” and “Asian/ Pacific Islander”, can be further investigated due to high “False stop” proportion. However, the Cramer’s V value, which was used to eliminate the influence of sample size, indicated that there was minor association between races and “False stop”.

Another testing part was “Level of Force” used among racial groups. Figure 26 illustrates percentages of each “Level of Force” based on racial groups. Since there were some differences among the groups, Kruskal Wallis H test was conducted to analyze the significance of those differences. The results are shown in Table 18.



**Figure 26** Percentages of “Level of Force” for each racial group

**Table 18** Results of hypothesis testing for “Level of Force” among racial groups

Variable	Value
H	183.80
Df	7
p-value	0.00
Alpha	0.05
Result	Reject null hypothesis

From Table 18, the conclusion is that there was a significant difference for “Level of Force” used among racial groups at 95% confidence level. According to Figure 26, “Black Hispanic”, “White Hispanic” and “Black” had high proportion of non-gunfire weapons used by police (Level 1). However, those groups also had high proportion of “Guilty” cases (low “False stop” proportion) regarding Figure 25. “Missing” group had high proportion of gunfire weapon used by police (Level 2), it suggests that further investigation can be done, due to lack of clear racial information.

To summarize, even though race was not the main driving factor of the predictions, there was significantly difference in SQF practices among the racial groups. Thus, NYPD should not overlook the racial issue and proceed the operation with caution and fairness.

## 4.3 Results of Super Learners

### 4.3.1 Guilty Prediction

In order to be used as base learners in experiment A2 and A4, single Logistic Regression and single Gaussian Naïve Bayes were optimized by parameter tuning. During the Super Learners constructing process, meta models were also optimized by parameters and threshold tuning. The ranges and optimum values for each technique are demonstrated in Appendix.

After tuning the parameters and threshold, cross-validation was conducted. The average results from 4-fold cross-validation of base models and Super Learners of all experiments are shown in Table 19 – 20. Base models were evaluated on the validation set before using meta models. Super Learners were evaluated on the validation set after using meta models. Bold numbers are the best results in each experiment.

**Table 19** Average results from 4-fold cross-validation of the base models

Base models		Metrics (4 fold cross validation)	
		Accuracy	F1 Score
<u>A1</u> Without tuning	DT	0.776	0.522
	RF	<b>0.844</b>	0.560
	XGB	0.798	<b>0.607</b>
<u>A2</u> Tuned	DT	0.745	0.555
	RF	0.824	0.621
	XGB	<b>0.848</b>	<b>0.660</b>
<u>A3</u> Without tuning	DT	0.776	0.521
	RF	<b>0.844</b>	0.560
	XGB	0.798	<b>0.607</b>
	LR	0.591	0.385
	GNB	0.707	0.343
<u>A4</u> Tuned	DT	0.745	0.555
	RF	0.824	0.621
	XGB	<b>0.848</b>	<b>0.660</b>
	LR	0.736	0.536
	GNB	0.764	0.511

\*DT = Decision Tree, RF = Random Forest, XGB = XGBoost, LR = Logistic Regression, GNB = Gaussian Naïve Bayes

**Table 20** Average results from 4-fold cross-validation of the Super Learners

Experiment	Meta model (optimal)	Metrics (4 fold cross validation)					Most important base model
		AUC of PRC	Accuracy	F1 Score	Precision	Recall	
<u>A1</u> (Base models: DT, RF, XGB without tuning)	DT	0.681	0.820	0.641	0.602	<b>0.687</b>	RF
	RF	0.685	0.831	0.645	0.632	0.659	RF
	XGB	0.700	0.825	0.645	0.615	0.679	RF
	LR	<b>0.714</b>	0.832	<b>0.646</b>	0.636	0.683	RF
	MLP	0.713	<b>0.838</b>	<b>0.646</b>	<b>0.660</b>	0.634	-
<u>A2</u> (Base models: DT, RF, XGB tuned)	DT	0.694	0.826	0.656	0.611	<b>0.709</b>	XGB
	RF	0.695	0.833	0.658	0.630	0.689	XGB
	XGB	<b>0.730</b>	0.838	0.659	0.647	0.672	XGB
	LR	0.729	<b>0.840</b>	<b>0.660</b>	<b>0.657</b>	0.663	XGB
	MLP	<b>0.730</b>	0.837	0.659	0.645	0.674	-
<u>A3</u> (Base models: DT, RF, XGB, LR, GNB without tuning)	DT	0.680	0.820	0.640	0.601	<b>0.686</b>	RF
	RF	0.713	0.825	0.646	0.614	0.681	RF
	XGB	0.714	0.823	0.644	0.609	0.683	DT
	LR	0.714	0.824	0.646	0.609	<b>0.686</b>	RF
	MLP	<b>0.717</b>	<b>0.828</b>	<b>0.648</b>	<b>0.624</b>	0.675	-
<u>A4</u> (Base models: DT, RF, XGB, LR, GNB tuned)	DT	0.680	0.820	0.640	0.601	0.686	XGB
	RF	0.690	0.833	0.657	0.631	<b>0.687</b>	XGB
	XGB	<b>0.730</b>	0.834	0.658	0.635	0.683	XGB
	LR	0.729	<b>0.840</b>	<b>0.660</b>	<b>0.656</b>	0.664	XGB
	MLP	<b>0.730</b>	0.839	<b>0.660</b>	0.654	0.666	-

\*DT = Decision Tree, RF = Random Forest, XGB = XGBoost, LR = Logistic Regression, GNB = Gaussian Naïve Bayes, MLP = Multilayer Perceptron Neural Network



XGBoost, Logistic Regression and Multilayer Perceptron (MLP) performed good as meta model across all experiments. Experiment A2 and A4 had better performance from tuned base models. XGBoost, Logistic Regression and MLP in both A2 and A4 showed similar performance. The best result among all was MLP in A4 (highlighted in Table 20), which reached 73.0% AUC and 66.0% F1 score. However, according to Table 8, the best Super Learner performance still did not reach the best single XGBoost performance (73.1% AUC, 66.2% F1 score).

In order to improve Super Learner performance, additional experiments were done. Even though experiment A2 and experiment A4 performed better, the performance did not improve from tuned base models shown in Table 19. And when adding more tuned base models, the performance of A4 did not improve from A2. Meanwhile, experiment A1 and experiment A3 significantly improved performance from their untuned base models, from F1 score 60.7% to 64.6% in experiment A1, and to 64.8% in experiment A3. Adding more untuned base models, in this case Logistic Regression and Gaussian Naïve Bayes, had potential to improve the performance.

So, the additional experiment A5 and experiment A6 were created as shown in Table 21. A5 focused on adding advanced tree-based models like XGBoost, in order to have base models with good performance. And A6 focused on adding simple classification models, in order to have base models with various structures and small run time. For meta model, XGBoost, Logistic Regression, and MLP were selected as they had good performance in previous experiments. The ranges and optimum values for each technique are demonstrated in Appendix. The performance of base models and Super Learners in experiment A5 and experiment A6 are shown in Table 22 – 23.

**Table 21** Additional Super Learner experimental plan (Guilty Prediction)

Prediction	Experiment	Base models	Meta model
Guilty Prediction	A5	DT, RF, XGB, BRF, AB, BG, ET, GBDT, HGB (No tuning)	XGB, LR, MLP (3 meta models for each experiment)
	A6	DT, LR, GNB, MNB, KNN (No tuning)	

\*DT = Decision Tree, RF = Random Forest, XGB = XGBoost, BRF = Balanced Random Forest, AB = AdaBoost, BG = Bagging, ET = Extra Trees, GBDT = Gradient Boosting Decision Tree, HGB = HistGradientBoost, LR = Logistic Regression, GNB = Gaussian Naïve Bayes, MNB = Multinomial Naïve Bayes, KNN = K Nearest Neighbor, MLP = Multilayer Perceptron Neural Network

**Table 22** Results from 4-fold cross-validation of the base models (additional experiments)

Base models		Metrics (4 fold cross validation)	
		Accuracy	F1 Score
<u>A5</u> without tuning	DT	0.776	0.521
	RF	0.844	0.600
	XGB	0.800	0.607
	BRF	0.796	<b>0.622</b>
	AB	0.810	0.546
	BG	0.837	0.559
	ET	<b>0.846</b>	0.579
	GB	0.826	0.495
	HGB	0.834	0.539
<u>A6</u> without tuning	DT	<b>0.776</b>	<b>0.521</b>
	LR	0.591	0.385
	GNB	0.707	0.343
	MNB	0.591	0.387
	KNN	0.760	0.359

\*DT = Decision Tree, RF = Random Forest, XGB = XGBoost, BRF = Balanced Random Forest, AB = AdaBoost, BG = Bagging, ET = Extra Trees, GB = GradientBoost, HGB = HistGradientBoost, LR = Logistic Regression, GNB = Gaussian Naïve Bayes, MNB = Multinomial Naïve Bayes, KNN = K Nearest Neighbor

**Table 23** Average results from 4-fold cross-validation of the Super Learners (additional experiments)

Experiment	Meta model (optimal)	Metrics (4 fold cross validation)					Most important base model
		AUC of PRC	Accuracy	F1 Score	Precision	Recall	
A5 (Base models: DT, RF, XGB, BRF, AB, BG, ET, GB, HGB, without tuning)	XGB	<b>0.724</b>	0.838	<b>0.655</b>	0.651	<b>0.659</b>	RF
	LR	0.721	<b>0.844</b>	0.654	<b>0.680</b>	0.629	GB
	MLP	0.723	0.843	0.654	0.675	0.635	-
A6 (Base models: DT, LR, GNB, MNB, KNN without tuning)	XGB	<b>0.561</b>	0.766	<b>0.543</b>	0.499	0.595	DT
	LR	0.551	<b>0.768</b>	0.537	<b>0.503</b>	0.576	GNB
	MLP	0.558	0.745	0.540	0.467	<b>0.641</b>	-

\*DF = Decision Tree, RF = Random Forest, XGB = XGBoost, BRF = Balanced Random Forest, AB = AdaBoost, BG = Bagging, ET = Extra Trees, GB = GradientBoost, HGB = HistGradientBoost, LR = Logistic Regression, GNB = Gaussian Naïve Bayes, MNB = Multinomial Naïve Bayes, KNN = K Nearest Neighbor, MLP = Multilayer Perceptron Neural Network

All tree-based base models in A5 showed better performance compared to A3 (best 71.7% AUC and 64.8% F1 score). All meta models showed similar performance, the best among others was XGBoost (highlighted in Table 23) with 72.4% AUC and 65.5% F1 score. Meanwhile, all A6 experiments showed significantly worse performance compared to A3. The best meta model was also XGBoost, with 56.1% AUC and 54.3% F1 score.

Even though adding more advanced tree-based base models helped improve performance, it still did not reach the performance of A4 which used tuned base models (best at 73.1% AUC and 66.0% F1 score).

### 4.3.2 Level of Force Prediction

In order to be used as base learners in experiment B2 and B4, single Logistic Regression and single Gaussian Naïve Bayes were optimized by parameter tuning. During the Super Learners constructing process, meta models were also optimized by parameter tuning. The ranges and optimum values for each technique are demonstrated in Appendix.

After tuning the parameters, cross-validation was conducted. The average results from 4-fold cross-validation of base models and Super Learners of all experiments are shown in Table 24 – 25. Base models were evaluated on the validation set before using meta models. Super Learners were evaluated on the validation set after using meta models. Bold numbers are the best results in each experiment.

**Table 24** Results from 4-fold cross-validation of the base models

Base models		Metrics (4 fold cross validation)	
		Accuracy	F1 Score (Macro average)
<u>B1</u> without tuning	DT	0.774	0.240
	RF	<b>0.850</b>	0.118
	XGB	0.681	<b>0.280</b>
<u>B2</u> Tuned	DT	0.694	0.249
	RF	0.730	0.309
	XGB	<b>0.806</b>	<b>0.384</b>
<u>B3</u> without tuning	DT	0.773	0.240
	RF	<b>0.850</b>	0.117
	XGB	0.681	<b>0.280</b>
	LR	0.538	0.137
	GNB	0.583	0.147
<u>B4</u> Tuned	DT	0.694	0.249
	RF	0.730	0.310
	XGB	<b>0.806</b>	<b>0.384</b>
	LR	0.493	0.216
	GNB	0.519	0.179

\*DT = Decision Tree, RF = Random Forest, XGB = XGBoost, LR = Logistic Regression, GNB = Gaussian Naïve Bayes

**Table 25** Results from 4-fold cross-validation of the Super Learners

Experiment	Meta model (optimal)	Metrics (4 fold cross validation)						Most important base model
		Accuracy	F1 Score			Precision (Macro average)	Recall (Macro average)	
			Level 1	Level 2	Macro average			
<u>B1</u> (Base models: DT, RF, XGB without tuning)	DT	0.570	0.352	0.212	0.282	0.182	<b>0.644</b>	RF
	RF	0.735	<b>0.409</b>	0.275	0.342	0.261	0.497	XGB
	XGB	0.786	0.395	0.320	<b>0.357</b>	0.337	0.388	RF
	LR	0.810	0.402	0.302	0.352	0.331	0.406	RF
	MLP	<b>0.857</b>	0.234	<b>0.346</b>	0.290	<b>0.759</b>	0.180	-
<u>B2</u> (Base models: DT, RF, XGB tuned)	DT	0.559	0.360	0.173	0.267	0.176	<b>0.650</b>	XGB
	RF	0.754	<b>0.408</b>	0.294	0.351	0.285	0.461	XGB
	XGB	0.785	0.394	0.335	<b>0.365</b>	0.344	0.395	XGB
	LR	0.803	0.393	0.298	0.345	0.318	0.399	XGB
	MLP	<b>0.851</b>	0.224	<b>0.344</b>	0.284	<b>0.625</b>	0.185	-
<u>B3</u> (Base models: DT, RF, XGB, LR, GNB without tuning)	DT	0.572	0.356	0.200	0.278	0.181	<b>0.647</b>	RF
	RF	0.752	<b>0.416</b>	0.301	<b>0.358</b>	0.284	0.486	XGB
	XGB	0.763	0.406	0.294	0.350	0.290	0.442	RF
	LR	0.810	0.401	0.305	0.353	0.332	0.405	RF
	MLP	<b>0.854</b>	0.279	<b>0.347</b>	0.313	<b>0.656</b>	0.206	-
<u>B4</u> (Base models: DT, RF, XGB, LR, GNB tuned)	DT	0.579	0.364	0.185	0.274	0.180	<b>0.637</b>	XGB
	RF	0.713	0.339	0.270	0.304	0.220	0.339	XGB
	XGB	0.758	<b>0.402</b>	0.282	0.342	0.280	0.440	RF
	LR	0.803	0.391	0.299	<b>0.345</b>	0.318	0.401	XGB
	MLP	<b>0.854</b>	0.229	<b>0.359</b>	0.294	<b>0.671</b>	0.190	-

\*DT = Decision Tree, RF = Random Forest, XGB = XGBoost, LR = Logistic Regression, GNB = Gaussian Naive Bayes,

MLP = Multilayer Perceptron Neural Network

As best results (bold numbers) in all metrics were scattered among techniques, the macro average of F1 score was the focused metric. Therefore, Random Forest, XGBoost, and Logistic Regression performed well as meta models across all experiments. The best result among all was XGBoost in B2, which reached a 36.5% macro average F1 score. However, according to Table 12, the best Super Learner performance still did not reach the best single XGBoost performance (38.4% macro average F1 score).

As well as Guilty Prediction, additional experiments were done. The performance of experiments B2 and experiment B4 did not improve from the tuned base models shown in Table 24. Meanwhile, experiment B1 and experiment B3 significantly improved performance from their untuned base models, from a macro average F1 score of 28.0% to 35.7% in experiment B1, and to 35.8% in experiment B3. Using untuned base models had the potential to improve the performance of Super Learners.

So, the additional experiment B5 and experiment B6 were created as shown in Table 26. B5 focused on adding advanced tree-based models like XGBoost, in order to have base models with good performance. And B6 focused on adding simple classification models, in order to have base models with various structures and small run times. For meta model, Random Forest, XGBoost, and Logistic Regression were selected as they had good performance in previous experiments. The performance of base models and Super Learners in experiment B5 and experiment B6 are shown in Table 27 – 28.

**Table 26** Additional Super Learner experimental plan (Level of Force Prediction)

Prediction	Experiment	Base models	Meta model
Level of Force Prediction	B5	DT, RF, XGB, BRF, AB, BG, ET, GB, HGB (No tuning)	RF, XGB, LR (3 meta models for each experiment)
	B6	DT, LR, GNB, MNB, KNN	

\*DT = Decision Tree, RF = Random Forest, XGB = XGBoost, BRF = Balanced Random Forest, AB = AdaBoost, BG = Bagging, ET = Extra Trees, GB = GradientBoost, HGB = HistGradientBoost, LR = Logistic Regression, GNB = Gaussian Naïve Bayes, MNB = Multinomial Naïve Bayes, KNN = K Nearest Neighbor

**Table 27** Average result from 4-fold cross-validation of the base models (additional experiments)

Base models		Metrics (4 fold cross validation)	
		Accuracy	F1 Score (Macro average)
<u>B5</u> without tuning	DT	0.773	0.239
	RF	0.850	0.117
	XGB	0.681	<b>0.280</b>
	BRF	0.524	0.242
	AB	0.853	0.192
	BG	0.848	0.184
	ET	<b>0.854</b>	0.191
	GB	0.843	0.030
	HGB	0.845	0.083
<u>B6</u> without tuning	DT	0.774	<b>0.241</b>
	LR	0.538	0.137
	GNB	0.583	0.147
	MNB	0.547	0.141
	KNN	<b>0.825</b>	0.106

\*DT = Decision Tree, RF = Random Forest, XGB = XGBoost, BRF = Balanced Random Forest, AB = AdaBoost, BG = Bagging, ET = Extra Trees, GB = GradientBoost, HGB = HistGradientBoost, LR = Logistic Regression, GNB = Gaussian Naïve Bayes, MNB = Multinomial Naïve Bayes, KNN = K Nearest Neighbor

**Table 28** Average results from 4-fold cross-validation of the Super Learners (additional experiments)

Experiment	Meta model (optimal)	Metrics (4 fold cross validation)						Most important base model
		Accuracy	F1 Score			Precision (Macro average)	Recall (Macro average)	
			Level 1	Level 2	Macro average			
B5 (Base models: DT, RF, XGB, BRF, AB, BG, ET, GB, HGB, without tuning)	RF	0.785	<b>0.427</b>	0.332	0.379	0.331	<b>0.446</b>	BRF
	XGB	0.785	0.419	<b>0.347</b>	<b>0.383</b>	<b>0.346</b>	0.436	BRF
	LR	<b>0.813</b>	0.406	0.297	0.351	0.335	0.404	XGB
B6 (Base models: DT, LR, GNB, MNB, KNN without tuning)	RF	0.682	0.310	<b>0.207</b>	<b>0.259</b>	0.196	0.384	GNB
	XGB	<b>0.698</b>	0.304	<b>0.207</b>	0.255	<b>0.199</b>	0.361	KNN
	LR	0.592	<b>0.312</b>	0.153	0.232	0.159	<b>0.499</b>	GNB

\*DT = Decision Tree, RF = Random Forest, XGB = XGBoost, BRF = Balanced Random Forest, AB = AdaBoost, BG = Bagging, ET = Extra Trees, GB = GradientBoost, HGB = HistGradientBoost, LR = Logistic Regression, GNB = Gaussian Naïve Bayes, MNB = Multinomial Naïve Bayes, KNN = K Nearest Neighbor

All tree-based base models in B5 showed better performance compared to B1-B4 (best 36.5% macro average F1 score). All meta models showed similar performance, the best among others was XGBoost (highlighted in Table 28) with a 38.3% macro F1 score. Meanwhile, all B6 experiments showed significantly worse performance compared to B1-B4. The best meta model was Random Forest, with a 25.9% macro average F1 score.



Unlike Guilty Prediction, the performance of B5, which added advanced tree-based base models, surpassed B1-B4. And the performance almost reached the single XGBoost performance (38.4% macro F1 score)

### 4.3.3 Confusion Matrix observation

Since F1 score, which is the focused metric in this study, is calculated from precision and recall, it is not clearly shown how precision and recall were changed through the process. Thus, the Confusion Matrix of some models were discussed in this section, in order to clearly see the change when the F1 score increased or decreased.

#### 4.3.3.1 Guilty Prediction

Confusion Matrix of the best untuned and tuned base model, XGBoost, are shown in Table 29 and Table 30, respectively. The result was obtained from testing trained models with the validation set.

**Table 29** Confusion matrix and related metrics of single XGBoost (untuned)

	Predicted Not Guilty	Predicted Guilty	
Actual Not Guilty	14994	2772	
Actual Guilty	1875	3674	
Class	Precision	Recall	F1 score
Not Guilty	0.89	0.84	0.87
Guilty	0.57	0.66	0.61

**Table 30** Confusion matrix and related metrics of single XGBoost (tuned)

	Predicted Not Guilty	Predicted Guilty	
Actual Not Guilty	16001	1865	
Actual Guilty	1821	3628	
Class	Precision	Recall	F1 score
Not Guilty	0.90	0.90	0.90
Guilty	0.66	0.67	0.66

From Table 29 and Table 30, after the XGBoost was tuned it predicted “Not Guilty” class more accurately. This reduced False Positive and significantly increased precision.

As mentioned in the previous section, Super Learners which used untuned base models can improve the performance from their base models. Confusion Matrix of the best Super Learner from Experiment A1 is shown in Table 31 as an example. For Super Learner which used tuned base models, the result is omitted as it is similar to the tuned base model (Table 30).

**Table 31** Confusion matrix and related metrics of Super Learner in Experiment A1 (Untuned tree-based models as base models, Logistic Regression as meta model)

	Predicted Not Guilty	Predicted Guilty	
Actual Not Guilty	15809	2057	
Actual Guilty	1872	3577	
Class	Precision	Recall	F1 score
Not Guilty	0.89	0.88	0.89
Guilty	0.63	0.66	0.65

From Table 31, the meta model predicted “Not Guilty” class more accurately than the untuned base models (Table 29). This also increased precision but still not as much as the tuned base model (Table 30). Both tuning the model and using a meta model used the same approach for increasing the F1 score, which is predicting the majority class more accurately.

#### 4.3.3.2 Level of Force Prediction

Confusion Matrix of the best untuned and tuned base model, XGBoost, are shown in Table 32 and Table 33, respectively. The result was obtained from testing trained models with the validation set.

**Table 32** Confusion matrix and related metrics of single XGBoost (untuned)

	Predicted Level 0	Predicted Level 1	Predicted Level 2
Actual Level 0	13730	3492	2126
Actual Level 1	1875	3674	383
Actual Level 2	145	108	322
Class	Precision	Recall	F1 score
Level 0	0.91	0.71	0.80
Level 1	0.29	0.48	0.36
Level 2	0.11	0.56	0.19

**Table 33** Confusion matrix and related metrics of single XGBoost (tuned)

	Predicted Level 0	Predicted Level 1	Predicted Level 2
Actual Level 0	16957	2099	248
Actual Level 1	1638	1348	54
Actual Level 2	286	98	198
Class	Precision	Recall	F1 score
Level 0	0.90	0.88	0.89
Level 1	0.38	0.44	0.41
Level 2	0.40	0.34	0.37

From Table 32 and Table 33, after the XGBoost was tuned the precision of both “Level 1” and “Level 2” increased, while recall decreased. This was caused by predicting fewer “Level 1” and “Level 2”, so it significantly reduced samples that were wrongly predicted but also reduced samples that were correctly predicted.

As mentioned in the previous section, Super Learners which used untuned base models can improve the performance from their base models. Confusion Matrix of the best Super Learner from Experiment B1 and B3 are shown in Table 34 and Table 35, respectively.

**Table 34** Confusion matrix and related metrics of Super Learner in Experiment B1 (untuned tree-based models as base models, XGBoost as meta model)

	Predicted Level 0	Predicted Level 1	Predicted Level 2
Actual Level 0	16395	2554	355
Actual Level 1	1513	1452	75
Actual Level 2	294	111	177
Class	Precision	Recall	F1 score
Level 0	0.90	0.85	0.87
Level 1	0.35	0.48	0.41
Level 2	0.29	0.30	0.30

**Table 35** Confusion matrix and related metrics of Super Learner in Experiment B3 (untuned tree-based, Logistic Regression, and Gaussian Naïve Bayes as base models, Random Forest as meta model)

	Predicted Level 0	Predicted Level 1	Predicted Level 2
Actual Level 0	15198	3396	710
Actual Level 1	1127	1768	145
Actual Level 2	212	140	230
Class	Precision	Recall	F1 score
Level 0	0.92	0.79	0.85
Level 1	0.33	0.58	0.42
Level 2	0.21	0.40	0.28

From Table 34, the meta model made the precision of both “Level 1” and “Level 2” increase, while recall of “Level 2” decreased compared to their untuned base models (Table 32). This was caused by predicting fewer “Level 1” and “Level 2”, so it significantly reduced samples that were wrongly predicted but also reduced samples that were correctly predicted. While in Table 35, the meta model made the recall of “Level 1” and “Level 2” high, while precision decreased compared to their untuned base model (Table 32). This was caused by predicting more “Level 1” and “Level 2”, so it significantly increased samples that were correctly predicted but also increased samples that were wrongly predicted. This demonstrates that two Super

Learners with different base learners and meta learner have different approaches to increase F1 score.

Additionally, from the result in the previous section, Super Learners using tuned base learners got a lower F1 score than their base models. To investigate the issue, Confusion Matrix of the best Super Learner from Experiment B2 and B4 are shown in Table 36 and Table 37, respectively.

**Table 36** Confusion matrix and related metrics of Super Learner in Experiment B2 (tuned tree-based models as base models, XGBoost as meta model)

	Predicted Level 0	Predicted Level 1	Predicted Level 2
Actual Level 0	16344	2675	285
Actual Level 1	1533	1452	55
Actual Level 2	282	111	189
Class	Precision	Recall	F1 score
Level 0	0.90	0.85	0.87
Level 1	0.34	0.48	0.40
Level 2	0.36	0.32	0.34

**Table 37** Confusion matrix and related metrics of Super Learner in Experiment B4 (tuned tree-based, Logistic Regression, and Gaussian Naïve Bayes as base models, Logistic Regression as meta model)

	Predicted Level 0	Predicted Level 1	Predicted Level 2
Actual Level 0	17026	1603	675
Actual Level 1	1714	1174	152
Actual Level 2	283	61	238
Class	Precision	Recall	F1 score
Level 0	0.90	0.88	0.89
Level 1	0.41	0.39	0.40
Level 2	0.22	0.41	0.29

From Table 36, the meta model predicted many samples as “Level 1” but they were actually “Level 0”, this made the precision of both “Level 1” and “Level 2” significantly

decreased from its tuned base model (Table 33), while the recall of “Level 1” increased and “Level 2” decreased. While in Table 37, the meta model predicted many samples as “Level 2” but they were actually “Level 0”, this made the precision “Level 2” significantly decreased from its tuned base model (Table 33), while the recall of “Level 2” increased. The meta model also predicted fewer samples as “Level 1”, so it reduced samples that were correctly predicted but also reduced samples that were wrongly predicted. This caused the precision of “Level 1” higher and the recall of “Level 1” lower, compared to its tuned base models (Table 33).

To summarize, it was found that Super Learners with untuned base models increased F1 score by 2 approaches; one is predicting the majority class more accurately, which caused higher precision, and another one is predicting the minority class more, which caused higher recall. On the other hand, the reason that some Super Learners had lower F1 score than their base models was the Super Learner predicted more samples as minority class but inaccurately, this significantly decreased precision.

## 4.4 Discussion of Super Learners

### 4.4.1 Base models

According to Table 20 and Table 25, when focusing on the F1 score, tuned base models performed better than untuned ones for Guilty Prediction, but for some experiments in Level of Force Prediction, the results were worse when using tuned base models.

Even though most experiments performed better when using tuned base models, they did not improve from the performance of their own base models. On the other hand, when using untuned base models, Super Learners could improve performance from their own base models. This happened in both Guilty and Level of Force Prediction. It can be implied that well-groomed base models may not be suitable for Super Learners. However, as Super Learners performed just almost as good as the single best XGBoost, it is questionable whether using tuned base models really did not cause better performance or the performance could not be any better as it had reached the limitations of the data itself.

Adding Logistic Regression and Gaussian Naïve Bayes as base models also had mixed results in both predictions. None of both models were important base models in any experiment, it can be implied that those two models had little impact on Super Learners. From Table 19 and

Table 24, base models with high accuracy or high F1 score were the most significant base model for all experiments. Because adding low-performance models may not be any help, the additional experiments were created to investigate more about adding various base models.

The results of additional experiments showed that the performance of base models significantly influenced the performance of meta models. Having high-performance base models improved meta models' performance and vice versa. Another interesting observation was the structure similarity of base models did not seem to impact the performance. Using various types of base models but with low accuracy, as in experiment A6 and experiment B6, did not help boost the performance, as well as adding different types of base models from A1/B1 to A3/B3.

#### 4.4.2 Meta models

For Guilty Prediction, XGBoost, Logistic Regression, and MLP had high AUC, Accuracy, and F1 score across all experiments. But for Level of Force Prediction, which was a multi-class classification, MLP did not have a high macro average F1 score. It was Random Forest that had a high macro average F1 score, although it had lower accuracy. Due to multiple classes and highly imbalance in Level of Force, MLP which cannot determine class weight may probably struggle to have high F1 score for minority classes.

So, the meta models that performed well across experiments were XGBoost and Logistic Regression. XGBoost has an advanced way of building trees that is probably the major cause of high predictive power as discussed in 4.2.2. On the other hand, Logistic Regression, which is one of the simple classification models, had a competitive performance as meta models. It is noteworthy that Logistic Regression had a significantly worse performance as base models compared to XGBoost but had close performance to XGBoost as meta models. It is possibly due to different kinds of problems in the input data between base models and meta models. The data input in base models were the SQF datasets, which had high numbers of features and required complex strategies to predict the results. Meanwhile, the data input in meta models were probabilities, only a few columns, and the goal was to find the best combination of those probabilities from base models. With the scheme of Logistic Regression that can be written as a formula of variables with coefficients, it might be more suitable for finding the best coefficients as weights for each probability of each base model, than solving the original predictions as base models.

#### 4.4.3 Summarized findings for structural designs of Super Learners

Summarized findings for structural designs of Super Learners are demonstrated in Table 38 for base models and Table 39 for meta models. Please note that the evaluation metrics that define how well the performance was, were AUC for Guilty Prediction and macro average F1 score for Level of Force Prediction.

**Table 38** Summarized findings for structural designs of Super Learners (base models)

Base model strategies	Not tuned	Tuned
Tree-based	<ul style="list-style-type: none"> <li>- Improve from base models</li> <li>- Worse than tuned</li> </ul>	<ul style="list-style-type: none"> <li>- Not improve from base models</li> <li>- Better than not tuned</li> </ul>
Tree-based with additional simple models	<ul style="list-style-type: none"> <li>- Improve from base models</li> <li>- Worse than tuned in Guilty Prediction and better in Level of Force Prediction</li> <li>- Improve from only tree-based</li> </ul>	<ul style="list-style-type: none"> <li>- Not Improve from base models</li> <li>- Better than tuned in Guilty Prediction and worse in Level of Force Prediction</li> <li>- Equal and worse performance from only tree-based</li> </ul>
Several advanced tree-based	<ul style="list-style-type: none"> <li>- Improve from base models</li> <li>- Best among all untuned base models</li> <li>- Competitive performance to tuned base models and single XGBoost</li> </ul>	-
Several various simple models	<ul style="list-style-type: none"> <li>- Improve from base models</li> <li>- Worst among all untuned base models</li> </ul>	-



**Table 39** Summarized findings for structural designs of Super Learners (meta models)

Meta model	Guilty Prediction	Level of Force Prediction
Decision Tree	- Worst performance - High recall	- Worst performance - High recall
Random Forest	- Competitive performance	- Good performance - High F1 score for Level 1
XGBoost	- Good performance	- Good performance - Worse performance when adding Logistic Regression and Gaussian Naïve Bayes to tree-based base models
Logistic Regression	- Good performance - High recall with untuned base models and high precision with tuned base models	- Good performance - Worse performance with tuned base models than untuned
MLP	- Good performance - High Precision	- High accuracy and Precision - High F1 score for Level 2 but Low for Level 1

Super Learners have competitive performance but still could not reach the best performance of a single XGBoost. Despite that, the main advantage of Super Learner found in this study is less effort used for optimizing models, in terms of computation time and resources. Logistic Regression is a simple model with a small run time, and the optimization process takes less effort and run time due to few parameters. Thus, in the case of Super Learner with untuned base models and Logistic Regression as a meta model, the performance comparison between optimized single XGBoost and the Super Learner is shown in Table 40.

**Table 40** Performance comparison between optimized single XGBoost and Super Learner with Logistic Regression as meta model

Prediction	Model		Metric	
	Base model	Meta model	F1 score	Accuracy
Guilty	Single XGBoost (untuned)		0.613	0.801
	Single XGBoost (tuned)		0.662	0.841
	DT, RF, XGB, BRF,	LR (tuned)	0.654	0.844
	AB, BG, ET, GB, HGB, (without tuning)	LR (without tuning)	0.648	0.811
Level of Force (F1 score is macro average for Level 1 and 2)	Single XGBoost (untuned)		0.275	0.676
	Single XGBoost (tuned)		0.384	0.806
	DT, RF, XGB, BRF,	LR (tuned)	0.351	0.813
	AB, BG, ET, GB, HGB, (without tuning)	LR (without tuning)	0.307	0.653

\*DT = Decision Tree, RF = Random Forest, XGB = XGBoost, BRF = Balanced Random Forest, AB = AdaBoost, BG = Bagging, ET = Extra Trees, GB = GradientBoost, HGB = HistGradientBoost, LR = Logistic Regression

Optimizing a single XGBoost takes a certain time and resources, on the other hand, using Super Learner with untuned base models and a simple meta model like Logistic Regression does not require such effort. From Table 40, for Guilty Prediction, simply using the Super Learner with untuned Logistic Regression as meta model obtained competitive performance compared to a single XGBoost. In this case, the Super Learner can be ready for implementation without an optimization process. For Level of Force Prediction, even though the Super Learner with untuned Logistic Regression as meta model did not have competitive performance, using the Super Learner with tuned Logistic Regression as meta model still takes less effort in the optimization process than a single XGBoost model.

For further study, Super Learner could be implemented for different kinds of problems, such as regression. Other base models and meta models could be explored. Also, the different structures of Super Learner could be investigated and improved, such as the number of folds for training base models.

## Chapter 5 : Conclusion

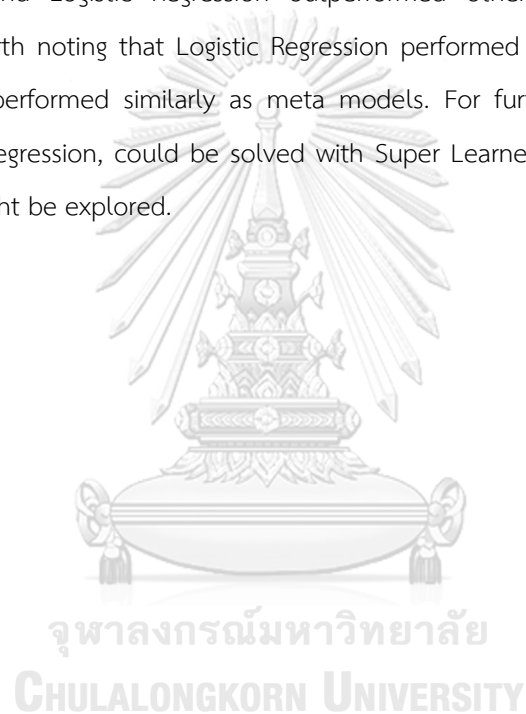
The objective of this study is to analyze the appropriateness of the NYPD officer's behavior in SQF practices. To achieve the objective, predictive models were created. Factors, relating to a stop that resulted in a conviction (Guilty Prediction) and police's weapon usage during a stop (Level of Force Prediction), were investigated using tree-based machine learning techniques. XGBoost models outperformed other techniques in both predictions. The performance of Guilty Prediction was at 65.9% F1 score and 84.0% accuracy. For Level of Force Prediction, the F1 score obtained for "Level 1" and "Level 2" were 40.7% and 35.0% respectively, with 80.4% overall accuracy.

The main findings are as follows. First, later years (close to 2019) led to more possibility for a case to be "Guilty" and also led to "Level 2" of Force usage. This indicates a tendency of fewer unnecessary stops but more police's weapons usage over time. Second, carrying weapons or carrying suspicious objects were the suspect's actions which led to "Guilty" cases. This suggests that weapons carriage was still a major element of criminal in the city. However, when "Criminal Possession of Weapon" was the suspected crime, the result led to "Not Guilty" cases yet "Level 2" of force usage. Taken together, these findings indicate a lack of accurate assumption about the suspect's weapons possession, which could be a major cause of unnecessary stops and overuse of force toward innocent citizens. Lastly, some suspect's actions led to "Level 2" of Force usage but did not lead to "Guilty" cases, and vice versa. This conflict showed that some protocols of the operation may be adjusted for appropriateness. Even though race was not the main driving factor of the predictions, there was significant different treatment in SQF practices among the racial groups. This suggests that NYPD should not overlook the racial issue and proceed the operation with caution and fairness. For further study, updated data could be gathered, and other actions that occurred during a stop, such as a frisk, might be explored to extend the perspective on the issue.

Additionally, this study also explored a hybrid technique called Super Learner. Experiments on various structures of Super Learners were done. The best model for Guilty Prediction used tuned classifiers including Decision Tree, Random Forest, XGBoost, Logistic Regression, and Gaussian Naïve Bayes, as base models, and used MLP as a meta model. It obtained 66.0% F1 score and 83.9% accuracy. And For Level of Force Prediction, the best model

used no-tuned various tree-based classifiers, such as XGBoost, Balanced Random Forest, as base models, and used XGBoost as a meta model. It obtained 41.9% F1 score and 34.7% F1 score for “Level 1” and “Level 2” respectively, with 78.5% accuracy.

The interesting findings of Super Learner structural designs are as follows. For base models, Super Learners could improve performance from their own base models when using untuned base models but did not improve when using tuned base models. The performance of base models also played a significant role in the performance of Super Learners, having high-performance base models improved meta models’ performance, and vice versa. For meta models, XGBoost and Logistic Regression outperformed other meta models across both predictions. It is worth noting that Logistic Regression performed much worse as base models than XGBoost but performed similarly as meta models. For further study, different kinds of problems, such as regression, could be solved with Super Learner. Also, other experiments on structural design might be explored.



## REFERENCES

- Agustika, D. K., Ariyanti, N. A., Wardana, I. N. K., Iliescu, D. D., & Leeson, M. S. (2021, 20-21 Oct. 2021). Classification of Chili Plant Origin by Using Multilayer Perceptron Neural Network. 2021 8th International Conference on Electrical Engineering, Computer Science and Informatics (EECSI),
- Akbaş, H., & Özdemir, G. (2020). An Integrated Prediction and Optimization Model of a Thermal Energy Production System in a Factory Producing Furniture Components. *Energies*, 13(22). <https://doi.org/10.3390/en13225999>
- Bacaksiz, E., & Koç, S. (2021). Comparison of Different Estimation Approaches in Rare Events Data. *Ege Akademik Bakis (Ege Academic Review)*, 263-272. <https://doi.org/10.21121/eab.960840>
- Brédy, J., Gallichand, J., Celicourt, P., & Gumiere, S. J. (2020). Water table depth forecasting in cranberry fields using two decision-tree-modeling approaches. *Agricultural Water Management*, 233, 106090. <https://doi.org/https://doi.org/10.1016/j.agwat.2020.106090>
- Chao, L., Wen-hui, Z., & Ji-ming, L. (2019). Study of Star/Galaxy Classification Based on the XGBoost Algorithm. *Chinese Astronomy and Astrophysics*, 43(4), 539-548. <https://doi.org/https://doi.org/10.1016/j.chinastron.2019.11.005>
- Chawla, N. V., Japkowicz, N., & Kotcz, A. (2004). Editorial: special issue on learning from imbalanced data sets. *SIGKDD Explor. Newsl.*, 6(1), 1-6. <https://doi.org/10.1145/1007730.1007733>
- Chen, T., & Guestrin, C. (2016). *XGBoost: A Scalable Tree Boosting System* Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, California, USA. <https://doi.org/10.1145/2939672.2939785>
- Coviello, D., & Persico, N. (2015). An Economic Analysis of Black-White Disparities in the New York Police Department's Stop-and-Frisk Program. *The Journal of Legal Studies*, 44(2), 315-360. <https://www.jstor.org/stable/26457029>
- Davis, J., & Goadrich, M. (2006). *The relationship between Precision-Recall and ROC*

- curves* Proceedings of the 23rd international conference on Machine learning, Pittsburgh, Pennsylvania, USA. <https://doi.org/10.1145/1143844.1143874>
- Ecevit, E. (2008). Usage of Penalized Maximum Likelihood Estimation Method in Medical Research: An Alternative to Maximum Likelihood Estimation Method. *Journal of Research in Medical Sciences*, 13.
- Fawcett, T. (2006). An introduction to ROC analysis. *Pattern Recognition Letters*, 27(8), 861-874. <https://doi.org/https://doi.org/10.1016/j.patrec.2005.10.010>
- Firth, D. (1993). Bias Reduction of Maximum Likelihood Estimates. *Biometrika*, 80(1), 27-38. <https://doi.org/10.2307/2336755>
- Freund, Y., & Schapire, R. E. (1997). A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting. *Journal of Computer and System Sciences*, 55(1), 119-139. <https://doi.org/https://doi.org/10.1006/jcss.1997.1504>
- Gelman, A., Fagan, J., & Kiss, A. (2007). An Analysis of the New York City Police Department's "Stop-and-Frisk" Policy in the Context of Claims of Racial Bias. *Journal of the American Statistical Association*, 102(479), 813-823. <https://doi.org/10.1198/016214506000001040>
- Geurts, P., Ernst, D., & Wehenkel, L. (2006). Extremely randomized trees. *Machine Learning*, 63(1), 3-42. <https://doi.org/10.1007/s10994-006-6226-1>
- Han, J., Kamber, M., & Pei, J. (2011). Data Mining: Concepts and Techniques Third Edition [M]. *The Morgan Kaufmann Series in Data Management Systems*, 5(4), 83-124. CHULALONGKORN UNIVERSITY
- Haykin, S. S. (2009). *Neural Networks and Learning Machines*. Pearson. <https://books.google.co.th/books?id=KCwW0AAACAAJ>
- Hedeya, M. A., Eid, A. H., & Abdel-Kader, R. F. (2020). A Super-Learner Ensemble of Deep Networks for Vehicle-Type Classification. *IEEE Access*, 8, 98266-98280. <https://doi.org/10.1109/ACCESS.2020.2997286>
- Helsby, J., Carton, S., Joseph, K., Mahmud, A., Park, Y., Navarrete, A., Ackermann, K., Walsh, J., Haynes, L., Cody, C., Patterson, M. E., & Ghani, R. (2017). Early Intervention Systems: Predicting Adverse Interactions Between Police and the Public. *Criminal Justice Policy Review*, 29(2), 190-209. <https://doi.org/10.1177/0887403417695380>

- Hu, X., Zhang, X., & Lovrich, N. (2021). Public perceptions of police behavior during traffic stops: logistic regression and machine learning approaches compared. *Journal of Computational Social Science*, 4(1), 355-380.  
<https://doi.org/10.1007/s42001-020-00079-4>
- Jerome, H. F. (2001). Greedy function approximation: A gradient boosting machine. *The Annals of Statistics*, 29(5), 1189-1232. <https://doi.org/10.1214/aos/1013203451>
- Jiang, H., Mao, H., Lu, H., Lin, P., Garry, W., Lu, H., Yang, G., Rainer, T. H., & Chen, X. (2021). Machine learning-based models to support decision-making in emergency department triage for patients with suspected cardiovascular disease. *International Journal of Medical Informatics*, 145, 104326.  
<https://doi.org/https://doi.org/10.1016/j.ijmedinf.2020.104326>
- Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., & Liu, T.-Y. (2017). *LightGBM: a highly efficient gradient boosting decision tree* Proceedings of the 31st International Conference on Neural Information Processing Systems, Long Beach, California, USA.
- Kibriya, A., Frank, E., Pfahringer, B., & Holmes, G. (2004). Multinomial naive Bayes for text categorization revisited. *Advances in Artificial Intelligence*, 488-499.
- Kobyliński, Ł., & Przepiórkowski, A. (2008). Definition Extraction with Balanced Random Forests. In B. Nordström & A. Ranta, *Advances in Natural Language Processing* Berlin, Heidelberg.
- Kotsiantis, S. B. (2013). Decision trees: a recent overview. *Artificial Intelligence Review*, 39(4), 261-283. <https://doi.org/10.1007/s10462-011-9272-4>
- Laan, M. J. v. d., Polley, E. C., & Hubbard, A. E. (2007). Super Learner. *Statistical Applications in Genetics and Molecular Biology*, 6(1).  
<https://doi.org/doi:10.2202/1544-6115.1309>
- Lin, N., Noe, D., & He, X. (2006). Tree-Based Methods and Their Applications. In H. Pham (Ed.), *Springer Handbook of Engineering Statistics* (pp. 551-570). Springer London.  
[https://doi.org/10.1007/978-1-84628-288-1\\_30](https://doi.org/10.1007/978-1-84628-288-1_30)
- Lundberg, S. M., Erion, G. G., & Lee, S.-I. (2018). Consistent Individualized Feature Attribution for Tree Ensembles. *ArXiv, abs/1802.03888*.
- MacDonald, J., & Braga, A. A. (2019). Did Post-Floyd et al. Reforms Reduce Racial

- Disparities in NYPD Stop, Question, and Frisk Practices? An Exploratory Analysis Using External and Internal Benchmarks. *Justice Quarterly*, 36(5), 954-983. <https://doi.org/10.1080/07418825.2018.1427278>
- Meng, Y., Yang, N., Qian, Z., & Zhang, G. (2021). What Makes an Online Review More Helpful: An Interpretation Framework Using XGBoost and SHAP Values. *Journal of Theoretical and Applied Electronic Commerce Research*, 16(3), 466-490. <https://www.mdpi.com/0718-1876/16/3/29>
- Milner, A. N., George, B. J., & Allison, D. B. (2016). Black and Hispanic Men Perceived to Be Large Are at Increased Risk for Police Frisk, Search, and Force. *PloS one*, 11(1), e0147158-e0147158. <https://doi.org/10.1371/journal.pone.0147158>
- Morrow, W. J., White, M. D., & Fradella, H. F. (2017). After the Stop: Exploring the Racial/Ethnic Disparities in Police Use of Force During Terry Stops. *Police Quarterly*, 20(4), 367-396. <https://doi.org/10.1177/1098611117708791>
- Neto, H. N. C., Lopez, M. A., Fernandes, N. C., & Mattos, D. M. F. (2020). MineCap: super incremental learning for detecting and blocking cryptocurrency mining on software-defined networking. *Annals of Telecommunications*, 75(3), 121-131. <https://doi.org/10.1007/s12243-019-00744-4>
- Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, É. (2011). Scikit-learn: Machine Learning in Python. *J. Mach. Learn. Res.*, 12(null), 2825–2830.
- Peterson, L. (2009). K-nearest neighbor. *Scholarpedia*, 4, 1883. <https://doi.org/10.4249/scholarpedia.1883>
- Polley, E. C., & Laan, M. J. v. d. (2010). Super Learner In Prediction.
- Saito, T., & Rehmsmeier, M. (2015). The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets. *PloS one*, 10(3), e0118432-e0118432. <https://doi.org/10.1371/journal.pone.0118432>
- van 't Wout, E., Pieringer, C., Torres Iribarra, D., Asahi, K., & Larroulet, P. (2021). Machine learning for policing: a case study on arrests in Chile. *Policing and Society*, 31(9), 1036-1050. <https://doi.org/10.1080/10439463.2020.1779270>
- Wei, N., Zhang, Q., Zhang, Y., Jin, J., Chang, J., Yang, Z., Ma, C., Jia, Z., Ren, C., Wu, L.,



Peng, J., & Mao, H. (2022). Super-learner model realizes the transient prediction of CO<sub>2</sub> and NO<sub>x</sub> of diesel trucks: Model development, evaluation and interpretation. *Environment International*, 158, 106977.

<https://doi.org/https://doi.org/10.1016/j.envint.2021.106977>

Wolpert, D. H. (1992). Stacked generalization. *Neural Networks*, 5(2), 241-259.

[https://doi.org/https://doi.org/10.1016/S0893-6080\(05\)80023-1](https://doi.org/https://doi.org/10.1016/S0893-6080(05)80023-1)

Yi, J., Zhang, H., Liu, H., Zhong, G., & Li, G. (2021). Flight Delay Classification Prediction Based on Stacking Algorithm. *Journal of Advanced Transportation*, 2021, 4292778. <https://doi.org/10.1155/2021/4292778>

Yu, H.-F., Huang, F.-L., & Lin, C.-J. (2011). Dual coordinate descent methods for logistic regression and maximum entropy models. *Machine Learning*, 85(1), 41-75.

<https://doi.org/10.1007/s10994-010-5221-8>

Zhang, H. (2004). The optimality of naive Bayes. *Aa*, 1(2), 3.

## APPENDIX

### Guilty Prediction (single model)

Ranges and optimum values of parameters and threshold for Decision Tree (Guilty Prediction)

Parameter/Threshold	Default	Range	Optimum Value
class_weight	'balanced' (manually set for imbalance data)		
criterion	'gini'	['gini', 'entropy']	'gini'
max_depth	None	[5, 10, 15]	10
min_samples_leaf	1	[1, 2, 4]	1
min_samples_split	2	[4, 6, 8, 10]	8
Threshold	0.5	0-1	0.585819

Ranges and optimum values of parameters and threshold for Random Forest (Guilty Prediction)

Parameter/Threshold	Default	Range	Optimum Value
bootstrap	True	[True,False]	False
class_weight	'balanced_subsample' (manually set for imbalance data)		
criterion	'gini'	['gini', 'entropy']	'entropy'
max_depth	None	[5, 10, 12, 15, 18, 20, 22, 25]	18
max_features	'sqrt'	['auto', 'sqrt', 'log2']	'auto'
min_samples_leaf	1	[1, 2, 4, 6, 8]	4
min_samples_split	2	[4, 6, 8, 10, 12]	8
n_estimators	100	[100, 200, 500, 800, 1000, 1400, 1500, 1600, 1800]	1600
Threshold	0.5	0-1	0.501037

Ranges and optimum values of parameters and threshold for XGBoost (Guilty Prediction)

Parameter/Threshold	Default	Range	Optimum Value
colsample_bylevel	1	[0.5, 0.6, 0.8, 1]	0.6
colsample_bynode	1	[0.5, 0.6, 0.8, 1]	0.6
colsample_bytree	1	[0.6, 0.8, 1]	0.6
gamma	0	[0, 0.1, 0.2, 0.3, 3, 5, 6, 7, 8, 10]	5
learning_rate	0.3	[0.01, 0.02, 0.03, 0.04, 0.05, 0.07, 0.08, 0.1]	0.01
max_depth	6	[10, 15, 18, 20, 22, 25, 28, 30]	25
min_child_weight	1	[1, 2, 4, 6, 8]	1
n_estimators	100	1-1974	1874
reg_alpha	0	[0, 0.1, 0.2, 0.3, 0.4, 0.5]	0.2
reg_lambda	1	[0.4, 0.6, 0.8, 1]	0.6
scale_pos_weight		3 (manually set for imbalance data)	
subsample	1	[0.6, 0.8, 1]	0.8
Threshold	0.5	0-1	0.444231

Ranges and optimum values of parameters and threshold for Logistic Regression (Guilty Prediction)

Parameter/Threshold	Default	Range	Optimum value
C	1	[0.01, 0.1, 1, 10, 100]	100
class_weight		'balanced' (manually set for imbalance data)	
penalty	'l2'	['none', 'l1', 'l2', 'elasticnet']	'l2'
solver	'lbfgs'	['newton-cg', 'lbfgs', 'liblinear', 'sag', 'saga']	'newton-cg'
Threshold	0.5	0-1	0.510124

Ranges and optimum values of parameters and threshold for Gaussian Naïve Bayes (Guilty Prediction)

Parameter/Threshold	Default	Range	Optimum value
Var_smoothing	1e-9	[1e-2, 1e-3, 1e-4, 1e-5, 1e-6, 1e-7, 1e-8, 1e-9, 1e-10, 1e-11, 1e-12, 1e-13, 1e-14, 1e-15]	1e-11
Threshold	0.5	0-1	0.679297

### Level of Force Prediction (single model)

Ranges and optimum values of parameters for Decision Tree (Level of Force Prediction)

Parameter	Default	Range	Optimum Value
class_weight		'balanced' (manually set for imbalance data)	
criterion	'gini'	['gini', 'entropy']	'gini'
max_depth	None	[15,18,20,22,25,28,30]	20
min_samples_leaf	1	[1,2,3,4,5]	1
min_samples_split	2	[2,3,4,5,6]	3

Ranges and optimum values of parameters for Random Forest (Level of Force Prediction)

Parameter	Default	Range	Optimum Value
bootstrap	True	[True,False]	False
class_weight		'balanced_subsample' (manually set for imbalance data)	
criterion	'gini'	['gini', 'entropy']	'gini'
max_depth	None	[8, 10, 11, 12, 13, 15, 18, 20, 22, 25]	15
max_features	'sqrt'	['auto', 'sqrt', 'log2']	'auto'
min_samples_leaf	1	[1, 2, 4, 6, 8, 10]	2
min_samples_split	2	[2, 4, 5, 8, 10, 12, 16, 20]	16
n_estimators	100	[100, 300, 400, 500, 600, 700, 800, 1000, 1200, 1400, 1500, 1600, 1800, 2000, 2200, 2500]	2000

Ranges and optimum values of parameters for XGBoost (Level of Force Prediction)

Parameter	Default	Range	Optimum Value
colsample_bylevel	1	[0.3, 0.4, 0.5, 0.6, 0.8, 1]	0.4
colsample_bynode	1	[0.6, 0.8, 1]	1
colsample_bytree	1	[0.2, 0.3, 0.4, 0.5, 0.6, 0.8, 1]	0.6
gamma	0	[0, 0.1, 0.2, 0.3, 0.5, 0.7, 1]	0.1
learning_rate	0.3	[0.03, 0.05, 0.07, 0.1, 0.3]	0.03
max_depth	6	[6, 8, 10, 12, 15, 20]	8
min_child_weight	1	[1, 2, 4, 6, 8, 10, 12]	8
n_estimators	100	1-2669	2569
num_class	3 (manually set from number of class in dataset)		
objective	multi : softmax (manually set for multi-class classification)		
reg_alpha	0	[0, 0.2, 0.4]	0.2
reg_lambda	1	[0.6, 0.8, 1]	0.8
subsample	1	[0.6, 0.8, 1]	0.8

Ranges and optimum values of parameters and threshold for Logistic Regression (Level of Force Prediction)

Parameter	Default	Range	Optimum value
C	1	[0.01, 0.1, 1, 10, 100]	1
class_weight	'balanced' (manually set for imbalance data)		
penalty	'l2'	['none', 'l1', 'l2', 'elasticnet']	'l2'
solver	'lbfgs'	['newton-cg', 'lbfgs', 'liblinear', 'sag', 'saga']	'newton-cg'

Ranges and optimum values of parameters and threshold for Gaussian Naïve Bayes (Level of Force Prediction)

Parameter	Default	Range	Optimum value
Var_smoothing	1e-9	[1e-2, 1e-3, 1e-4, 1e-5, 1e-6, 1e-7, 1e-8, 1e-9, 1e-10, 1e-11, 1e-12, 1e-13, 1e-14, 1e-15]	1e-10

### Guilty Prediction (Super Learner Experiment A1-A4)

Ranges and optimum values of parameters and threshold for Decision Tree as meta model  
(Guilty Prediction)

Parameter/Threshold	Default	Range	Optimum Value for each experiment			
			A1	A2	A3	A4
class_weight	'balanced' (manually set for imbalance data)					
criterion	'gini'	['gini', 'entropy']	'entropy'	'entropy'	'entropy'	'entropy'
max_depth	None	[1, 2, 3, 5, 8, 10, 15]	1	1	1	1
min_samples_leaf	1	[1, 2, 4]	1	1	1	1
min_samples_split	2	[2, 4, 6, 8]	2	2	2	2
Threshold	0.5	0-1	0.5	0.5	0.5	0.5

Ranges and optimum values of parameters and threshold for Random Forest as meta model  
(Guilty Prediction)

Parameter /Threshold	Default	Range	Optimum Value for each experiment			
			A1	A2	A3	A4
bootstrap	True	[True,False]	True	True	True	True
class_weight	'balanced_subsample' (manually set for imbalance data)					
criterion	'gini'	['gini', 'entropy']	'entropy'	'entropy'	'entropy'	'entropy'
max_depth	None	[1, 3, 5, 8, 10, 15]	1	1	5	1
max_features	'sqrt'	['auto', 'sqrt', 'log2']	'auto'	'auto'	'auto'	'sqrt'
min_samples_leaf	1	[1, 2, 4, 6, 8, 10]	1	2	4	8
min_samples_split	2	[2, 5, 8, 10, 12]	2	2	10	8
n_estimators	100	[100, 200, 500, 800, 1000, 1200, 1500]	200	1000	1000	500
Threshold	0.5	0-1	0.601776	0.507906	0.568302	0.568302

Ranges and optimum values of parameters and threshold for XGBoost as meta model (Guilty Prediction)

Parameter/Threshold	Default	Range	Optimum Value for each experiment			
			A1	A2	A3	A4
colsample_bylevel	1	[0.2, 0.3, 0.4, 0.5, 0.6, 0.8, 1]	0.4	1	0.6	0.6
colsample_bynode	1	[0.6, 0.8, 1]	1	1	0.6	1
colsample_bytree	1	[0.6, 0.8, 1]	1	1	0.8	1
gamma	0	[0, 0.1, 1, 3]	0	0	5	1
learning_rate	0.3	[0.01, 0.02, 0.03, 0.05, 0.07, 0.1]	0.01	0.07	0.05	0.07
max_depth	6	[1,2,3,5]	1	1	3	1
min_child_weight	1	[1, 4, 8]	1	1	1	4
n_estimators	100	[60, 80, 100, 200, 300, 500, 1000]	100	500	100	500
reg_alpha	0	[0, 0.2, 0.4]	0	0	0	0
reg_lambda	1	[0.6, 0.8, 1]	1	1	0.8	1
scale_pos_weight		3 (manually set for imbalance data)				
subsample	1	[0.6, 0.8, 1]	1	1	0.6	1
Threshold	0.5	0-1	0.525549	0.581517	0.54389	0.562639

Ranges and optimum values of parameters and threshold for Logistic Regression as meta model  
(Guilty Prediction)

Parameter/Threshold	Default	Range	Optimum Value for each experiment			
			A1	A2	A3	A4
C	1	[0.01, 0.1, 1, 10, 100]	1	0.01	1	0.01
class_weight	'balanced' (manually set for imbalance data)					
penalty	'l2'	['none', 'l1', 'l2', 'elasticnet']	'l2'	'l1'	'l2'	'l1'
solver	'lbfgs'	['newton-cg', 'lbfgs', 'liblinear', 'sag', 'saga']	'lbfgs'	'saga'	'lbfgs'	'saga'
Threshold	0.5	0-1	0.572728	0.606267	0.528609	0.604695



Ranges and optimum values of parameters and threshold for Multilayer Perceptron Neural Network as meta model (Guilty Prediction)

Parameter /Threshold	Default	Range	Optimum Value for each experiment			
			A1	A2	A3	A4
activation	'relu'	['identity', 'logistic', 'tanh', relu']	'relu'	'tanh'	'relu'	'logistic'
alpha	0.0001	[0.00001, 0.00005, 0.0001, 0.05]	0.0001	0.01	0.0001	0.00001
hidden_layer_sizes	(100,)	[(10,30,10), (20,), (40,), (40,40), (40,40,40,), (50,50,50), (50,100,50,), (60,), (80,), (100,), (100,100,100)]	(100,)	(10,30,10)	(50,100,50)	(40,)
learning_rate	'constant'	['constant', 'invscaling', 'adaptive']	'constant'	'constant'	'constant'	'constant'
random_state	99					
solver	'adam'	['lbfgs', 'sgd', 'adam']	'adam'	'sgd'	'adam'	'adam'
Threshold	0.5	0-1	0.342049	0.32137	0.289899	0.335143

### Level of Force Prediction (Super Learner Experiment A1-A4)

Ranges and optimum values of parameters and threshold for Decision Tree as meta model

(Level of Force Prediction)

Parameter	Default	Range	Optimum Value for each experiment			
			B1	B2	B3	B4
class_weight	'balanced' (manually set for imbalance data)					
criterion	'gini'	['gini', 'entropy']	'gini'	'gini'	'gini'	'entropy'
max_depth	None	[1, 2, 3, 5, 8, 10, 15]	2	2	2	3
min_samples_leaf	1	[1, 2, 4]	1	1	1	1
min_samples_split	2	[2, 4, 6, 8]	2	2	2	2

Ranges and optimum values of parameters and threshold for Random Forest as meta model

(Level of Force Prediction)

Parameter	Default	Range	Optimum Value for each experiment			
			B1	B2	B3	B4
bootstrap	True	[True,False]	True	True	True	True
class_weight	'balanced_subsample' (manually set for imbalance data)					
criterion	'gini'	['gini', 'entropy']	'gini'	'gini'	'gini'	'gini'
max_depth	None	[1, 3, 5, 8, 10, 15, 18, 20]	15	15	15	15
max_features	'sqrt'	['auto', 'sqrt', 'log2']	'auto'	'log2'	'log2'	'sqrt'
min_samples_leaf	1	[1, 2, 4, 6, 8, 10]	4	4	4	4
min_samples_split	2	[2, 5, 8, 10, 12]	10	8	5	2
n_estimators	100	[100, 200, 500, 800, 1000, 1200, 1500, 1800]	200	500	1500	500

Ranges and optimum values of parameters and threshold for XGBoost as meta model (Level of Force Prediction)

Parameter	Default	Range	Optimum Value for each experiment			
			B1	B2	B3	B4
colsample_bylevel	1	[0.4, 0.5, 0.6, 0.8, 1]	0.8	0.8	0.8	0.5
colsample_bynode	1	[0.4, 0.5, 0.6, 0.8, 1]	1	1	0.5	0.4
colsample_bytree	1	[0.2, 0.3, 0.4, 0.5, 0.6, 0.8, 1]	0.5	0.5	0.4	0.5
gamma	0	[0, 0.1, 0.5, 1, 1.5, 3, 5, 7]	1	1	1	1
learning_rate	0.3	[0.03, 0.05, 0.07, 0.1, 0.2, 0.3]	0.05	0.05	0.1	0.1
max_depth	6	[1, 5, 8, 10, 12, 15, 18]	15	15	10	10
min_child_weight	1	[1, 4, 8]	1	4	1	1
n_estimators	100	[100, 200, 300, 500, 800, 1000]	500	500	300	300
num_class	3 (manually set from number of class in dataset)					
objective	multi : softmax (manually set for multi-class classification)					
reg_alpha	0	[0, 0.2, 0.4]	0	0	0.2	0
reg_lambda	1	[0.6, 0.8, 1]	0.8	0.8	0.8	1
subsample	1	[0.6, 0.8, 1]	0.8	0.8	1	0.8

Ranges and optimum values of parameters and threshold for Logistic Regression as meta model  
(Level of Force Prediction)

Parameter	Default	Range	Optimum Value for each experiment			
			B1	B2	B3	B4
C	1	[0.01, 0.1, 1, 10, 100]	1	0.01	10	0.01
class_weight	'balanced' (manually set for imbalance data)					
penalty	'l2'	['none', 'l1', 'l2', 'elasticnet']	'l2'	'l1'	'l2'	'l1'
solver	'lbfgs'	['newton-cg', 'lbfgs', 'liblinear', 'sag', 'saga']	'liblinear'	'liblinear'	'liblinear'	'liblinear'

Ranges and optimum values of parameters and threshold for Multilayer Perceptron Neural Network as meta model (Level of Force Prediction)

Parameter	Default	Range	Optimum Value for each experiment			
			B1	B2	B3	B4
activation	'relu'	['identity', 'logistic', 'tanh', relu']	'relu'	'tanh'	'tanh'	'relu'
alpha	0.0001	[0.00001, 0.00005, 0.0001, 0.05]	0.0001	0.0001	0.0001	0.0001
hidden_layer _sizes	(100,)	[(10,30,10), (20,),(40,), (40,40), (40,40,40,), (50,50,50), (50,100,50,), (60,),(80,), (100,), (100,100,100) ]	(100,)	(50,50,50)	(100,200,1 00)	(100,)
learning_rate	'constant'	['constant', 'invscaling', 'adaptive']	'constant'	'constant'	'constant'	'constant'
random_stat e	99					
solver	'adam'	['lbfgs', 'sgd', 'adam']	'adam'	'sgd'	'adam'	'adam'

### Guilty Prediction (Super Learner Experiment A5-A6)

Ranges and optimum values of parameters and threshold for XGBoost as meta model in additional experiments (Guilty Prediction)

Parameter/Threshold	Default	Range	Optimum Value for each experiment	
			A5	A6
colsample_bylevel	1	[0.6, 0.8, 1]	0.6	1
colsample_bynode	1	[0.6, 0.8, 1]	0.6	1
colsample_bytree	1	[0.6, 0.8, 1]	0.6	1
gamma	0	[0, 0.1, 1, 3, 5, 7, 10]	0	7
learning_rate	0.3	[0.01, 0.02, 0.03, 0.05, 0.07, 0.1, 0.3]	0.05	0..02
max_depth	6	[1, 2, 3, 5, 7, 10]	7	3
min_child_weight	1	[1, 4, 8]	1	1
n_estimators	100	[60, 80, 100, 200, 300, 500]	100	300
reg_alpha	0	[0, 0.2, 0.4, 1, 3]	0.2	0
reg_lambda	1	[0, 0.6, 0.8, 1, 3]	0.8	1
scale_pos_weight	3 (manually set for imbalance data)			
subsample	1	[0.4, 0.5, 0.6, 0.8, 1]	1	0.5
Threshold	0.5	0-1	0.581434	0.522632

Ranges and optimum values of parameters and threshold for Logistic Regression as meta model in additional experiments (Guilty Prediction)

Parameter/Threshold	Default	Range	Optimum Value for each experiment	
			A5	A6
C	1	[0.01, 0.1, 1, 10, 100]	1	0.01
class_weight	'balanced' (manually set for imbalance data)			
penalty	'l2'	['none', 'l1', 'l2', 'elasticnet']	'l2'	'l2'
solver	'lbfgs'	['newton-cg', 'lbfgs', 'liblinear', 'sag', 'saga']	'lbfgs'	'liblinear'
Threshold	0.5	0-1	0.639204	0.570276

Ranges and optimum values of parameters and threshold for Multilayer Perceptron Neural Network as meta model in additional experiments (Guilty Prediction)

Parameter /Threshold	Default	Range	Optimum Value for each experiment	
			A5	A6
activation	'relu'	['identity', 'logistic', 'tanh', relu']	'logistic'	'relu'
alpha	0.0001	[0.0001, 0.001, 0.01, 0.05]	0.0001	0.0001
hidden_layer_sizes	(100,)	[(30,30,30), (50,), (50,50), (50,50,50), (50,100,50,), (60,), (80,), (80,80), (80,80,80), (100,), (100,100,100)]	(80,)	(50,50,50)
learning_rate	'constant'	['constant', 'invscaling', 'adaptive']	'constant'	'constant'
random_state		99		
solver	'adam'	['lbfgs', 'sgd', 'adam']	'adam'	'adam'
Threshold	0.5	0-1	0.351285	0.240983

#### Level of Force Prediction (Super Learner Experiment A5-A6)

Ranges and optimum values of parameters and threshold for Random Forest as meta model in additional experiments (Level of Force Prediction)

Parameter	Default	Range	Optimum Value for each experiment	
			B5	B6
bootstrap	True	[True,False]	True	True
class_weight		'balanced_subsample' (manually set for imbalance data)		
criterion	'gini'	['gini', 'entropy']	'entropy'	'gini'
max_depth	None	[5, 10, 11, 12, 13, 14, 15, 16, 17, 18, 20]	15	14
max_features	'sqrt'	['auto', 'sqrt', 'log2']	'auto'	'auto'
min_samples_leaf	1	[1, 2, 3, 4, 5, 6, 8]	3	4
min_samples_split	2	[2, 3, 4, 5, 6, 10]	5	5
n_estimators	100	[100, 200, 300, 400, 500, 600, 700, 800, 900]	700	600

Ranges and optimum values of parameters and threshold for XGBoost as meta model in additional experiments (Level of Force Prediction)

Parameter	Default	Range	Optimum Value for each experiment	
			B5	B6
colsample_bylevel	1	[0.4, 0.5, 0.6, 0.8, 1]	0.6	.0.6
colsample_bynode	1	[0.6, 0.8, 1]	1	1
colsample_bytree	1	[0.4, 0.5, 0.6, 0.8, 1]	0.6	1
gamma	0	[0, 0.5, 1, 1.5, 2, 3, 5, 7, 10]	1	3
learning_rate	0.3	[0.05, 0.07, 0.1, 0.3]	0.07	0.01
max_depth	6	[1, 5, 6, 7, 8, 10, 15]	10	8
min_child_weight	1	[1, 4, 8, 10, 20]	1	1
n_estimators	100	[100, 200, 300, 500, 800]	300	500
num_class	3 (manually set from number of class in dataset)			
objective	multi : softmax (manually set for multi-class classification)			
reg_alpha	0	[0, 0.2, 0.4, 0.5, 1, 5, 10]	1	0
reg_lambda	1	[0.5, 0.6, 1, 1.5, 3, 5, 10]	1	1
subsample	1	[0.6, 0.8, 1]	0.8	0.8

Ranges and optimum values of parameters and threshold for Logistic Regression as meta model in additional experiments (Level of Force Prediction)

Parameter	Default	Range	Optimum Value for each experiment	
			B5	B6
C	1	[0.01, 0.1, 1, 10, 100]	0.1	0.01
class_weight	'balanced' (manually set for imbalance data)			
penalty	'l2'	['none', 'l1', 'l2', 'elasticnet']	'l1'	'l2'
solver	'lbfgs'	['newton-cg', 'lbfgs', 'liblinear', 'sag', 'saga']	'liblinear'	'newton-cg'



## VITA

**NAME** Passiri Bodhidatta

**DATE OF BIRTH** 1 October 1998

**PLACE OF BIRTH** Bangkok

**INSTITUTIONS ATTENDED** Bachelor of Engineering (Industrial Engineering)  
Chulalongkorn University

**HOME ADDRESS** 440/2 Paholyothin rd. Samsennai Phayathai Bangkok  
10400

