Національний лісотехнічний університет України
(повне найменування вищого навчального закладу)

Навчально-науковий інститут коп'ютерних наук
та інформаційних технологій
(повне найменування інституту, назва факультету (відділення))

Кафедра комп'ютерних наук
(повна назва кафедри (предметної, циклової комісії))

Магістерська кваліфікаційна робота

другий (магістерський)
(рівень вищої освіти)

на тему: **Інтелектуальна система аналізу криміногенної ситуації**

Виконав: студент VI курсу, групи КН-61м
спеціальності 122 – "Комп'ютерні науки"
(шифр і назва напряму підготовки, спеціальності)
Кириленко В. С.
(прізвище та ініціали)

Керівник Шиманський В. М.
(прізвище та ініціали)

Рецензент _____
(прізвище та ініціали)

Львів – 2024 р.

Національний лісотехнічний університет України
(повне найменування вищого навчального закладу)

ННІ комп'ютерних наук та інформаційних технологій

Кафедра комп'ютерних наук

Рівень вищої освіти другий (магістерський)

Спеціальність 122 "Комп'ютерні науки"
(шифр і назва)

ЗАТВЕРДЖУЮ
Завідувач кафедри

_____ Борецька І. Б.
"____" _____2024 року

З А В Д А Н Н Я
НА ДИПЛОМНУ РОБОТУ СТУДЕНТУ

Кириленко Владислав Сергійович
(прізвище, ім'я, по батькові)

1. Тема роботи **Інтелектуальна система аналізу криміногенної ситуації**

керівник роботи Шиманський В. М, канд. фіз.-мат. наук, асистент
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджені наказом вищого навчального закладу від 13.02. 2023 року № С-49

2. Термін подання студентом роботи 05. 01. 2024 р.

3. Вихідні дані до роботи:

- вивчити предметну область, проаналізувати існуючі фактори та методи моделювання, а також відповідні програмні продукти;
- розглянути і використати алгоритми, які лежать в основі математичної моделі інформаційної системи;
- спроектувати інформаційну систему з допомогою мови програмування Python та відповідних бібліотек для побудови інтерфейсу та візуалізації результатів.
- представити результати роботи інформаційної системи.

4. Зміст пояснювальної записки (перелік питань, які потрібно розробити)

Розділ 1. Стан проблемної області

Розділ 2. Інформаційне забезпечення

Розділ 3. Математичне забезпечення

Розділ 4. Програмне забезпечення

Розділ 5. Стартап проекту

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)

Додаток А.

Додаток Б.

6. Дата видачі завдання 15 лютого 2023 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломної роботи	Строк виконання етапів роботи	Примітка
1	Огляд літературних даних та інших джерел згідно досліджуваної теми	15.02-31.03.2023	виконано
2	Аналіз досліджуваної теми та вибір відповідних варіантів її розробки	01.04-30.04. 2023	виконано
3	Постановка задачі та її формалізація	01.05-31.05. 2023	виконано
4	Вибір та обґрунтування методів і засобів проведення дослідження	01.06-30.06. 2023	виконано
5	Розроблення концептуальної схеми реалізації завдання	01.07-31.07. 2023	виконано
6	Програмна реалізація завдання	01.08-30.09. 2023	виконано
7	Тестування програмного продукту та отриманих результатів	01.10-30.10. 2023	виконано
8	Розробка пояснювальної записки магістерської роботи	01.11-30.11. 2023	виконано
9	Корегування пояснювальної записки згідно вимог, розроблення презентації	01.12-04.01. 2024	виконано

Студент _____ Кириленко В. С.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Шиманський В. М.
(підпис) (прізвище та ініціали)

РЕФЕРАТ

Дипломна робота містить 99 сторінок пояснювальної записки, 9 рисунків, 2 таблиці, 14 джерел, 2 додатки.

В роботі приводиться опис найбільш відомих моделей прогнозування, досліджується взаємозв'язок кількості правопорушень та різних зовнішніх факторів (дня тижня або погодних умов). На підставі отриманих даних приведено графіки, які відображають найбільш небезпечний час доби, дні тижня та місяці в році, в які необхідно посилити поліцейський контроль на вулицях міста. Окрім цього, розроблено інтерактивну карту, яка дозволить знайти найбільш небезпечні райони та вулиці міста. На основі отриманих закономірностей розроблено модель машинного навчання для прогнозування рівня правопорушень, які враховують як історичні дані, так і різні зовнішні фактори. Приводяться оцінки точності отриманих результатів, за якими можна зробити висновок про якість прогнозів.

В цій роботі отримано прогнози криміногенного рівня злочинності в місті Нью-Йорк, на основі яких можуть бути визначені несприятливі дні, в яких число правопорушень значно перевищує середній рівень.

Ключові слова: прогнозування рівня злочинності, Python, Pandas, Seaborn, Scikit-Learn, Folium.

ABSTRACT

Diploma paper contains 99 pages of explanatory note, 9 figures, 2 tables, 14 sources, 2 appendix.

The comprehensive course project provides a description of the most well-known forecasting models, examines the relationship between the number of offenses and various external factors (day of the week or weather conditions). Based on the received data, graphs are given that reflect the most dangerous time of the day, days of the week and months of the year, when it is necessary to strengthen police control on the city streets. In addition, an interactive map has been developed that will allow you to find the most dangerous areas and streets of the city. Based on the obtained regularities, a machine learning model was developed for forecasting the level of offenses, which takes into account both historical data and various external factors. Estimates of the accuracy of the obtained results are given, based on which a conclusion can be drawn about the quality of the forecasts.

In this work, forecasts of the criminogenic level of crime in the city of Chicago were obtained, on the basis of which unfavorable days can be determined, in which the number of offenses significantly exceeds the average level.

Keywords: Python, Pandas, Seaborn, Scikit-Learn, Folium.

ТЕХНІЧНЕ ЗАВДАННЯ

В дипломній роботі потрібно вирішити такі завдання. Розробити інтелектуальну систему для аналізу криміногенної ситуації.

1. Вивчити предметну область та провести аналіз програмних продуктів, призначених для аналізу, моніторингу та прогнозування рівня злочинності.

2. Розобити математичну модель даної інтелектуальної системи.

3. Розробити мовою програмування Python та відповідних бібліотек аналізу та візуалізації інтелектуальну систему аналізу криміногенної ситуації.

4. Преставити результати роботи даної системи, дослідити її параметри.

ПЕРЕЛІК СКОРОЧЕНЬ І СПЕЦІАЛЬНИХ ТЕРМІНІВ

- IoT – Internet of Things: концепція даних між фізичними об’єктами, які мають вбудовані засоби та технології для взаємодії одне з одним чи з зовнішнім середовищем;
- GPS – Global Positioning System: система глобального позиціонування;
- EBIT – Evidence Based Investigation Tool: система для отримання ймовірного прогнозу потенційного розкриття порушень деяких типів;
- Hart – інструмент оцінки ризиків;
- NDAS – National Data Analytics Solution: прогнозування потенційних злочинців та жертв тяжких насильницьких злочинів;
- PredPol – система прогнозування правопорушень;
- Precobs – Pre Crime Observation System: система прогнозування правопорушень;
- RMS – Report Management System;
- ГІС – геоінформаційні системи;
- МВС – міністерство внутрішніх справ;
- СППР – системи підтримки прийняття рішень;
- ШІ – штучний інтелект.

ЗМІСТ

ВСТУП

Актуальність дипломної роботи

На даний час досить гостро стоїть проблема безпеки, тому рівень злочинності є важливим фактором при виборі місця для переїзду, виборі житла в тому чи в іншому районі міста. На передбачення рівня правопорушень впливає багато факторів різного роду. Існують державні структури, які займаються пошуком та затримкою зловмисників, а також намагаються попередити злочини. У сучасному світі технологій набули розвитку системи відеонагляду, які дозволяють швидко знайти людину, яка здійснила злочин, а методи криміналістики практично не залишають злочинцеві шансів уникнути правосуддя. Але набагато краще, якщо завдяки дії правоохоронних органів правопорушення не відбулося б зовсім. Завдяки розвитку інформаційних технологій існують системи, які дозволяють передбачити виникнення правопорушень в тому або в іншому районі міста.

У даній роботі розроблена модель машинного навчання для передбачення правопорушень. В якості даних для побудови моделі використовуються реальні дані про злочини в місті Нью-Йорк. Розроблена модель дозволить помітити закономірності, які не так очевидні, дізнатися, коли, де і які правопорушення відбуваються частіше, тим самим, отримати модель, яка дозволить заздалегідь дізнатися про те, куди потрібно буде надіслати додатковий патруль для профілактики правопорушень.

Предметом дослідження є розробка інформаційно-аналітичної системи для аналізу правопорушень.

Об’єктом дослідження є криміногенна ситуація в м. Нью-Йорк.

Мета роботи – розробка та реалізація інформаційно-аналітичної системи для можливості аналізу та прогнозування рівня злочинності.

Завдання:

1. Вивчити предметну область та провести аналіз програмних продуктів, призначених для аналізу, моніторингу та прогнозування рівня злочинності.

2. Розобити математичну модель даної інтелектуальної системи.

3. Розробити мовою програмування Python та відповідних бібліотек аналізу та візуалізації інтелектуальну систему аналізу криміногенної ситуації.

4. Преставити результати роботи даної системи, дослідити її параметри.

Наукова новизна одержаних результатів

Наукова обгрунтованість положень та висновків підтверджується використанням великого об’єму даних спостережень. Отримано якісні прогнози рівня злочинності у місті Нью-Йорк, на підставі яких можуть бути визначені несприятливі дні, в які число злочинів значно перевищує середній рівень. Приведено опис найбільш відомих моделей прогнозування, досліджено взаємозв’язок кількості злочинів та різних зовнішніх факторів (дня тижня, погодних умов).

Практичне значення одержаних результатів

Результати, які отримані в рамках цієї роботи, можна використати для прогнозування рівня злочинності в інших містах. Дані методи прогнозування можуть бути корисні для зниження рівня злочинності, так як дозволяють передбачати несприятливі періоди, в які необхідно посилити патрулювання вулиць міста.

РОЗДІЛ 1. АНАЛІЗ СТАНУ ПРОБЛЕМНОЇ ОБЛАСТІ

1.1. Штучний інтелект як засіб прогнозування та протидії правопорушенням

Технології штучного інтелекту сьогодні є лідером у галузі цифрових телекомунікацій. Системи на основі штучного інтелекту спрямовані на активне застосування в криміналістиці, що є логічним продовженням процесу цифровізації та алгоритмізації розслідування правопорушень. Практична реалізація представленої концепції дасть інструмент, що об’єднює в собі об’єм даних, що володіє аналітичним функціоналом, що дозволяє налаштувати неявні зв’язки при розслідуванні багатьох правопорушень.

Черговим кроком у розвитку інформаційних систем МВС України могли б стати технології з елементами ШІ. Відмінність системи на основі штучного інтелекту від будь-якої традиційної інформаційної системи з заздалегідь визначеним алгоритмом дій, закладеним у її архітектурі, полягає в наступному. ШІ – система, яка здатна на основі вхідних даних і умов самостійно прийняти рішення, що відрізняється від заздалегідь визначеного алгоритму, або створити альтернативний алгоритм рішення задачі. Наприклад, часто при роботі автоматизованої системи дорожніх камер фіксації порушення швидкісного режиму помилково виписувався штраф водієві лише з урахуванням зв’язку з державним номерним знаком - власником автомашини тому, що така інформація міститься в базі даних. Система з ШІ в цьому випадку могла б на підставі невідповідності державного номерного знака і марки автомобіля, наприклад, відстежувати автомобілі з аналогічними або ідентичними номерами і на підставі цих даних приймати рішення про коректність фіксації.

Автоматизовані системи мають потенціал розвитку в напрямку ШІ, так як реалізовані в них функції розмітки слідів, перевірок, рекомендаційних списків запрограмовані і пусті та ефективні, але все це

алгоритм. Місцем застосування елементів ШІ тут могли б стати засоби постійного моніторингу та перевірки слідів, роботи з рекомендованими списками на рівні машинного навчання.

Процеси цифровізації торкаються не тільки технологічного сектору економіки та цифрових корпорацій, але й абсолютно різних сфер діяльності фізичних та юридичних осіб. Зафіксовано перехід технологій штучного інтелекту від реалізації пілотних проектів на новий етап розвитку: до широкомасштабного впровадження в технологічні процеси та виведення на ринок масових цифрових продуктів. Це веде за собою впровадження глобальних трендів цифровізації як у кримінальну сферу, так і в сферу протидії злочинній діяльності. Завдяки консервативності державних законодавчих механізмів маємо ситуацію, коли теоретичний фундамент юридичних наук не встигає за практичною реалізацією та впровадженням у криміналістику нових інформаційних цифрових технологій. У цей же час розвиток цифрових технологій рухається далі і ставить перед криміналістикою питання практичного застосування штучного інтелекту.

Технології штучного інтелекту в останні роки все частіше потрапляють у сферу уваги вчених-юристів. При цьому розглядаються як позитивні сторони, наприклад, реалізація прогностичних функцій у кримінологічних задачах, використання штучного інтелекту при отриманні та аналізі оперативно-пошукової інформації, так і негативні, які пов’язані з протидією загрозам, що базуються на тих же технологіях штучного інтелекту.

Розробка системи підтримки прийняття рішень (СППР) є логічним продовженням робіт з алгоритмізації та автоматизації методики розслідування правопорушень. Загальні питання якісного застосування комп’ютерної техніки як засобів криміналістичної техніки, прорив технологій в швидкості (передачі, обробки) та об’ємів інформації дозволили перейти на новий рівень обробки інформації – настала ера

великих даних Big Data. Великі дані – це не просто величезний об'єм структурованих і неструктурованих даних, це ще й різноманітний інструментарій, який призначений для їх обробки. В якості технологічних інструментів для обробки великих даних це машинне навчання, штучні нейронні мережі, розпізнавання образів, прогнозна аналітика, тобто це є ті інструменти, які зараз прийнято називати штучним інтелектом.

Одним із напрямків розвитку технологій штучного інтелекту є комп'ютерна лінгвістика. Наведена назва не містить у собі юридичної складової, комп'ютерна лінгвістика є науковим напрямком в області комп'ютерного та комп'ютерного моделювання інтелектуальних процесів у людини, що має на меті використання математичних моделей для опису природних мов. Завдяки своєму прикладному характеру результати роботи в області комп'ютерної лінгвістики використовуються в криміналістиці: це система розпізнавання тексту, аналіз інтернет-контенту з метою виділення певного змісту (екстремістського, терористичного характеру). В меншій мірі розроблено напрям створення онтології. В області інформатики створення онтології – це спроба формалізації певної області знань за допомогою концептуальної схеми. В якості однієї з таких областей може здійснюватися діяльність по розслідуванню правопорушень.

В перспективі в якості джерел даних можна використовувати такий величезний інформаційний ресурс, як криміналістичні обліки та інші бази даних МВС.

Машинне навчання – це один із напрямів штучного інтелекту, основний принцип якого полягає в тому, що машини отримують дані та навчаються на їх основі. В даний час це найбільш перспективний інструмент для бізнесу, науки і сфери прийняття управлінських рішень. Системи машинного навчання дозволяють швидко застосовувати знання, отримані при навчанні на великих наборах даних, за рахунок чого можуть розпізнавати особи, речі, об'єкти та багатьох інших. Глибоке навчання являється підмножиною машинного обучения. Воно використовує деякі

методи машинного навчання для вирішення реальних завдань, застосовуючи нейронні мережі, які можуть імітувати прийняття рішень людиною.

Відмінна особливість сучасної злочинності – її здатність брати на озброєння передові технологічні рішення, що обумовлено її безмежними фінансовими можливостями та певною прозорістю наукових досягнень. Вивчення сучасної високотехнологічної злочинності дозволяє виділити наступні способи і сфери використання зловмисниками штучного інтелекту.

Фішинг – отримання доступу до конфіденційного користувача, його логінів і паролів. Дрони стали застосовуватися для слідкування за агентами та співробітниками правоохоронних органів, за свідками правопорушень і членами конкуруючих злочинних угруповань, а також у пошуках об’єктів для грабежів і пограбувань, з метою слідкування за контейнерами з контрабандним товаром.

Спостерігається процес автоматизації соціальної інженерії, прикладом чого є так звані боти. Бот – це програма, яка здатна за певним алгоритмом виконувати будь-які дії через інтерфейси, призначені для людей, наприклад вести діалог з відвідувачами форуму або в соцмережі.

ІТ-технології можливо застосувати і для прогнозування злочинності. Дослідники проблем використання штучного інтелекту в кримінології приводять дані інтерполу, згідно з якими більш ніж у 70 країнах світу поліцейські на практиці використовують ці або інші дані прогностичної аналітики, спираючись на програмні засоби понад 25 корпорацій-виробників. Предиктивна або прогнозна аналітика являє собою сукупність методів аналізу даних, які спрямовані на прогнозування поведінки людей.

1.2. Системи предиктивної аналітики для попередження правопорушень

Програми передбачення злочинності (predictive policing) займаються збором і аналізом даних про правопорушення, що сталися, для того щоб визначити майбутнього порушника або передбачити місце злочину. Припускається, що це дозволить поліцейським попередити можливі події або ввести ті чи інші профілактичні заходи.

Сучасні програми можна розділити на два типи, які користуються популярністю в США та Західній Європі:

• person-based – орієнтовані на виявлення людини, яка ймовірно здійснить злочин;

• place-based – орієнтовані на місце, в якому можливо буде здійснено злочин.

Даних інструментів є дуже багато. Це може бути і попередження рецидивів після умовно-дострокового звільнення, аналіз мереж та відкритих соціальних даних у пошуках потенційних правонарушників, підрахунок суми застав і розміру наказання для засуджених, результатів передбачених для них ймовірності повторно порушити закон, багато іншого. Ці системи з’явилися в результаті того, що в поліцейських копіях було достатньо багато даних про громадян – після переведення документообороту на цифрові рельси їх можна було зібрати й проаналізувати.

Основним завданням поліцейських стає контроль гарячих точок – місць з непропорційно високою кількістю та інтенсивністю певних правопорушень. Через аналіз динаміки таких гарячих точок вимірювалась ефективність інструментів прогнозування злочинності. Якщо алгоритм вірно визначив, що в якому-небудь місці виникає хот-спот або навпаки, правильно визначив безпечні райони, то технологія вважалася ефективною.

Самі програми прогнозування місця правопорушень розвивалися від гнучких, але складних для роботи, до простих, Це добре ілюструє історія двох гігантів цього ринку – компаній PredPol і HunchLab.

HunchLab проектувався як багатофункціональний помічник поліцейського. Програма повинна була допомогти офіцеру проаналізувати різні види правопорушень і задіяти для цього максимально широкий інструментарій. Наприклад, технологія HunchLab надає карти прогнозованих показників залежно від вибраної статистичної моделі та тонкості підготовки даних для аналізу (поточного стану злочинності, погоди, соціально-економічних показників різних частин міста і тому подібного).

Модель PredPol була побудована на кардинально інших засадах – створена вона як система, яка була необхідною лише для даних про кримінальні інциденти. На їх основі вона видавала поліцейському карту, на якій підсвічувала місця, де ймовірність здійснення злочину виявилася високою, а йому було потрібно лише вирішити, як швидко об'їхати всі виділені точки. Ця модель швидко поширилася по відділах поліції США.

PredPol і аналогічні системи швидко набули популярності у правоохоронних органів США та інших країн, завдяки дослідженням, які підтвердили, що технологія зменшує кількість правопорушень. В цьому сенсі PredPol допоміг поліцейським в гарячих точках краще, ніж це практикувалося до цього людська аналітика поліцейських департаментів.

Одна з тих областей, в яких штучний інтелект демонструє здатність робити дуже точні прогнози, стала громадська безпека. Зараз у світі розгорнуті кілька проектів для прогнозування правопорушень, в тому числі, на основі аналізу великих даних і машинного навчання.

США. PredPol (Predictive Policing)

Це американська система прогнозування правопорушень, яка розроблена компанією The Predictive Policing Company. В даний час розгорнуто не менше 50 локальних систем прогнозування в департаментах

поліції США різних штатів. PredPol використовує алгоритм машинного навчання для створення прогнозів злочинів на основі трьох типів даних: вид порушення, місце порушення, дані та час порушення. На виході генерує відповідний прогноз: що станеться (вид порушення), де відбудеться (місце порушення) і коли відбудеться (дата / час порушення).

Прогнози відображаються у вигляді червоних прямокутників у веб-інтерфейсі з використанням картографічного інтерфейсу Google Maps. Кожний квадрат на карті відповідає реальній частині місцевості розмірами 150 × 150 метрів, він являє собою область найбільшого ризику здійснення правопорушення та актуалізується не тільки щодня, але й для кожної зміни поліції (денної, нічної). Офіцерам передбачається витрачати близько 10 % часу зміни (близько 6 хвилин на годину) на патрулювання областей, які вказані цією системою.

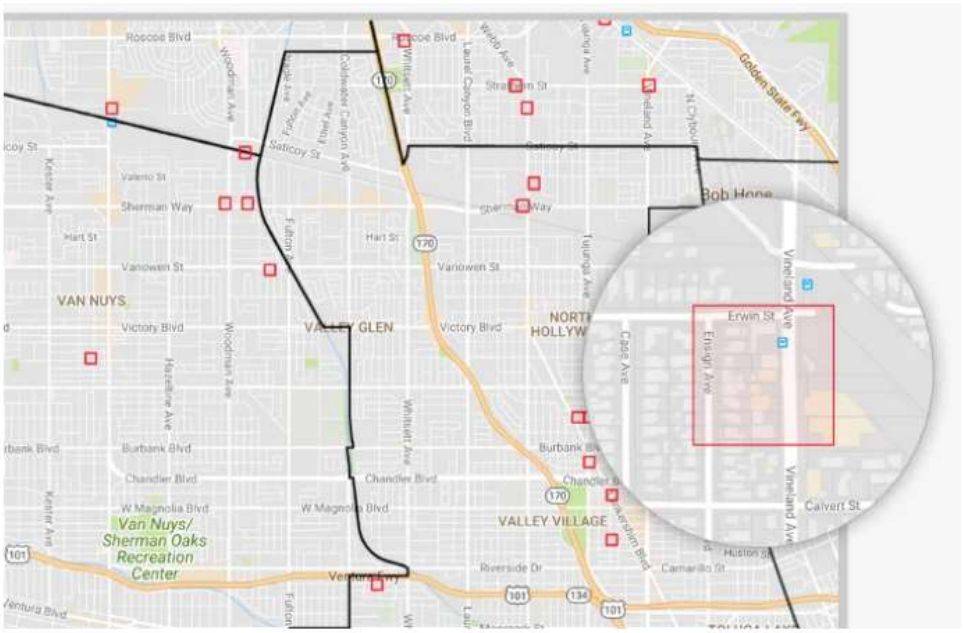


Рис. 1.1. Карта з прогнозом правопорушень PredPol.

Для машинного навчання використовуються набори даних кримінальної статистики за період від 2 до 5 років. При цьому для кожного міста повинна працювати своя окрема система, з поправкою на специфіку місцевих умов. Після першого запуску та початкового навчання система PredPol самостійно та щодня оновлює алгоритм прогнозування, на основі

даних про нові правопорушення, які завантажуються із системи управління звітами Report Management System (RMS) поліцейського департаменту.

Система дозволяє призначати екіпажам патрулювання окремих областей у конкретний час і автоматизувати моніторинг виконання завдань за допомогою системи позиціонування GPS. Статистика правопорушень і прогнозів злочинів візуалізується, є можливість створювати звіти за допомогою фільтрів (вид правопорушення, період).

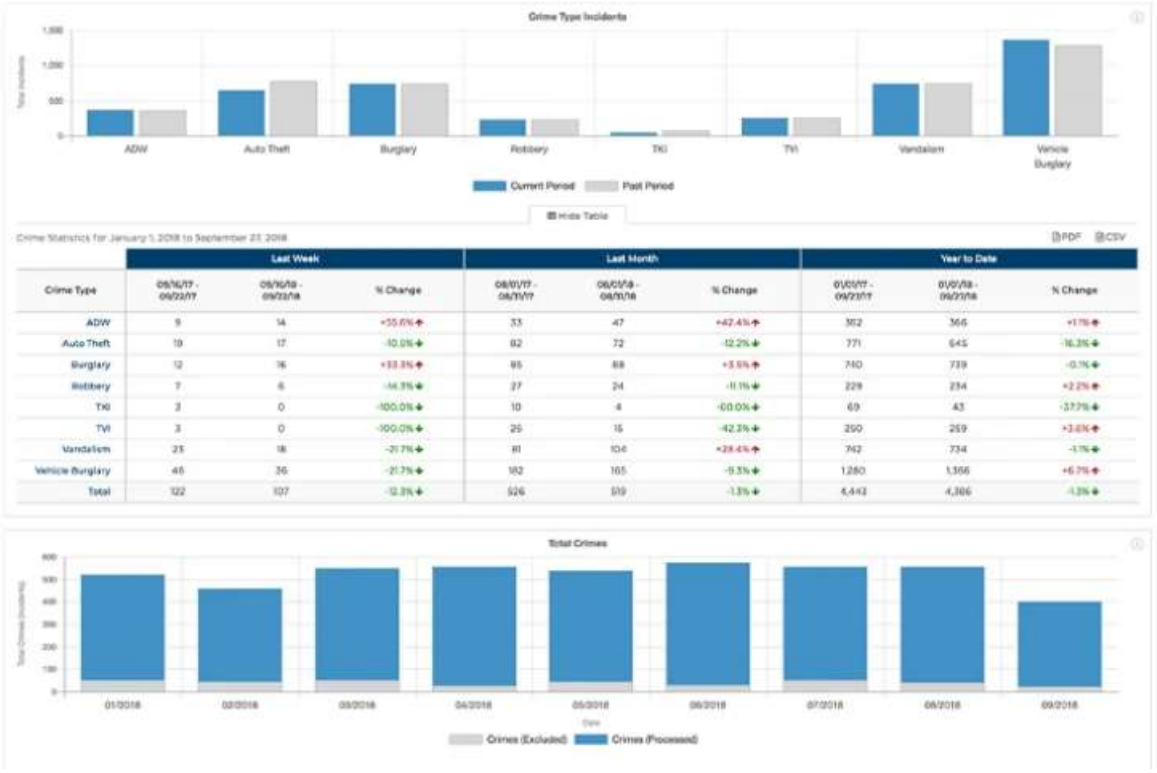


Рис. 1.2. Compstat – модуль візуалізації системи PredPol.

Алгоритм прогнозування правопорушень PredPol має наукове обґрунтування і витримав більш ніж 10-річну перевірку критикою та практичною роботою. У перших версіях системи прогнозування правопорушень була використана модель аналізу гарячих точок (hot-spot analysis). Однак пізніше було знайдено більш ефективне рішення.

Для цього в якості алгоритму машинного навчання для отримання прогнозів були застосовані нейронні мережі. Таке поєднання показало високу результативність для прогнозування правопорушень. Робота моделі може бути пояснена з позиції кримінологічних і соціологічних теорій.

Якщо будинок був пограбований сьогодні, ризик того, що він буде пограбований післязавтра, дійсно зростає. Це пов’язано з тим, що для злочинців раціонально повертатися туди, де вони досягли успіху раніше: не має сенсу йти в якийсь інший невідомий будинок, де вони нічого не знають про присутність людей, про складність проникнення і наявності охоронних систем, а також про можливу вигоду, тоді як про будинок, який вони пограбували два-три дні тому, вони знають дуже багато, і цей варіант набагато менш ризикованіший.

Але не тільки цей пограбований будинок піддається більшому ризику повторного пограбування, але і сусідній будинок також має великий ризик бути пограбованим, адже сусіди досить схожі: у них один соціально-економічний статус, вони працюють подібним чином, у них схожий будинок, і у них буде приблизно стільки же речей і цінностей, які можна вкрасти. Так що сценарій пограбування, який злочинець уже використовував, буде майже ідеально підходити для пограбування і сусіднього будинку.

В якомусь сенсі правопорушення – це всього лише фізичний процес, і якщо можна пояснити, що керується правопорушниками і як вони перетинаються зі своїми жертвами, машинне навчання дозволяє PredPol аналізувати дані, робити висновки та встановлювати зв’язок між великими об’ємами даних, з якими людська аналітика просто не може справитися. Машинне навчання надає набір підходів до виявлення статистичних закономірностей у даних, які складно описуються стандартними математичними моделями або виходять за рамки природних способів сприйняття людини-експерта.

На ринку представлено досить багато рішень: Crime Prediction and Prevention (IBM), CommandCentral Analytics (Motorola Solutions), Intelligence-led policing (Microsoft), HunchLab (Azavea), Vantara (Hitachi), SAS, Palantir.

Великобританія. Поліція англійського графства Кент була першим підрозділом правоохоронних органів Великобританії, яке використовувало в своїй роботі рішення для прогнозування порушень, яке базується на алгоритмах аналізу великих даних. При цьому поліція Великобританії вже досить давно застосовує більш прості рішення для прогнозної аналітики, наприклад, картування гарячих точок; прогнозні моделі випадків, які повторюються. Також в різні роки на практиці застосовувалися рішення типу прогнозної системи домашнього насилля RFG (Recency Frequency and Gravity system) або її більш розвиненої форми системи для контролю осіб, які уразливі ризику спричинення шкоди iVPD.

Що стосується розробок на основі штучного інтелекту та великих даних, то на сьогоднішній день відомо наступне. По-перше, поліція Лондона вже тестує систему прогнозування вдосконалення окремих видів правопорушень, які аналогічні до PredPol, але розроблені вже власними силами. Інше застосування ШІ – прогнозування потенційних злочинців та жертв тяжких насильницьких злочинів. Ця система носить назву NDAS (National Data Analytics Solution).

За допомогою NDAS британська поліція хоче отримувати прогнози щодо громадян, які ризикують здійснити правопорушення із застосуванням вогнепальної чи холодної зброї або стати жертвами таких злочинів, а також тих, хто може стати жертвами сучасної форми рабства. Прогнози генеруються на основі машинного навчання на основі інформації з різних місцевих і національних поліцейських баз даних, у тому числі звітів про правопорушення, записів про затримання, звіти про осіб, які пропали безвісті. Ці прогнози будуть використані не для арештів, а для раннього втручання, щоб допомогти утримати потенційних правонарушників від здійснення злочину або захистити потенційних жертв.

В NDAS може знайти своє застосування інша розробка – інструмент оцінки ризиків (Hart). Харт прогнозує ймовірність повторного

правопорушення особами, які були раніше засуджені. Рішення застосовує машинне навчання для визначення ймовірності повторного здійснення людиною правопорушення протягом наступних двох років на основі таких даних, як попередня кримінальна історія, вік і адреса проживання.

Деякі підрозділи правоохоронних органів Великобританії стали керуватися порадами ШІ при прийнятті рішень про те, чи доцільно братися за розслідування конкретного правопорушення, або це не має сенсу. Для цього використовують інструмент для розслідування, який базується на доказовості EBIT (Evidence Based Investigation Tool) – алгоритм для отримання ймовірного прогнозу потенційного розкриття порушень деяких типів (вуличні напади, порушення громадського порядку, які складають більшу частину зареєстрованих правопорушень).

З усієї множини зареєстрованих правопорушень ШІ вказує на те, які з найбільшою ймовірністю можуть бути розкриті, а також такі, для яких ймовірність успішного розслідування наближається до нуля.

В EBIT навчений алгоритм на тисячах нападів і порушень громадського порядку. Він визначив вісім факторів, які впливають на те, чи може бути правопорушення в кінцевому підсумку розслідувано: наявність свідків, записів з камери відеоспостереження, конкретних підозрюваних. Алгоритм прогнозу базується на минулих даних, але вся кількість факторів з часом може змінюватися, відповідно, в системі можуть виникати помилки. Для часткового вирівнювання цих помилок в EBIT закладений механізм сліпих тестів: алгоритм кожний день спеціально включає в число рекомендованих для розслідування 1-2 випадково обраних правопорушень з низькою ймовірністю розкриття.

Японія

В Японії була розгорнута система прогнозування правопорушень, яка була запущена до проведення Олімпійських ігор, які проходили в Токіо в 2020 році. Основана на штучному інтелекті прогнозна система використовує алгоритми глибокого аналізу великих даних і самонавчання.

Модель прогнозування правопорушень використовує кримінологічні, математичні та статистичні методи аналізу даних про час, місце, погоду, географічні умови та інші характеристики злочинів та інцидентів, а також інформацію із зовнішніх джерел. Розробкою системи займається компанія Singular Perturbations спільно з поліцією. Вона вже пробує адаптувати існуючі моделі та методи прогнозування правопорушень з врахуванням специфічних умов, які властиві для Японії.

Німеччина

В даний час поліцейські системи предиктивної аналітики використовуються в шести федеральних землях Німеччини. Основне призначення таких систем полягає у використанні статистичного аналізу для виявлення областей, в яких з найбільшою ймовірністю можуть статися пограбування квартир, офісів і автомобілів.

Найбільш відомою з них є система прогнозування правопорушень Precobs (Pre Crime Observation System), яка використовується не тільки в Німеччині, але і в Швейцарії. Крім того, в Німеччині діють такі місцеві системи прогнозування правопорушень, як KrimPro (Берлін), KLB-operativ (Хессен), PreMap (Нижня Саксонія), SKALA (Північний Рейн і Вестфалія).

В основі системи прогнозування правопорушень Precobs лежать кримінологічні теорії попередження, що й в американській системі Predictive Policing. Precobs збирає дані кримінальної статистики за 5-річний період (пов’язані з пограбуванням: місце злочину, спосіб злочину, вид вкраденого майна, тип будівлі). В цих даних алгоритм Precobs шукає і знайде такі закономірності, як переваги способів, шляху і час вдосконалення правопорушень, які людина не може побачити.

Спочатку визначаються критерії виявлення повторюваних правопорушень (тригерів), наприклад, місце порушення, спосіб порушення. Потім на основі аналізу визначаються області, в яких виявляються правопорушення з високою степінню подібності. Ці області стають просторовою основою моніторингу для створюваних прогнозів

правопорушень. Коректність вибраних критеріїв і визначення областей перевіряються методом комп'ютерного моделювання. Після цього генеруються тимчасові прогнози очікуваної реєстрації появлення тригера. Прогноз відображається у вигляді квадратів розміром 250 х 250 м або радіусом 500 м, різні кольори відображають ймовірність реалізації прогнозу.

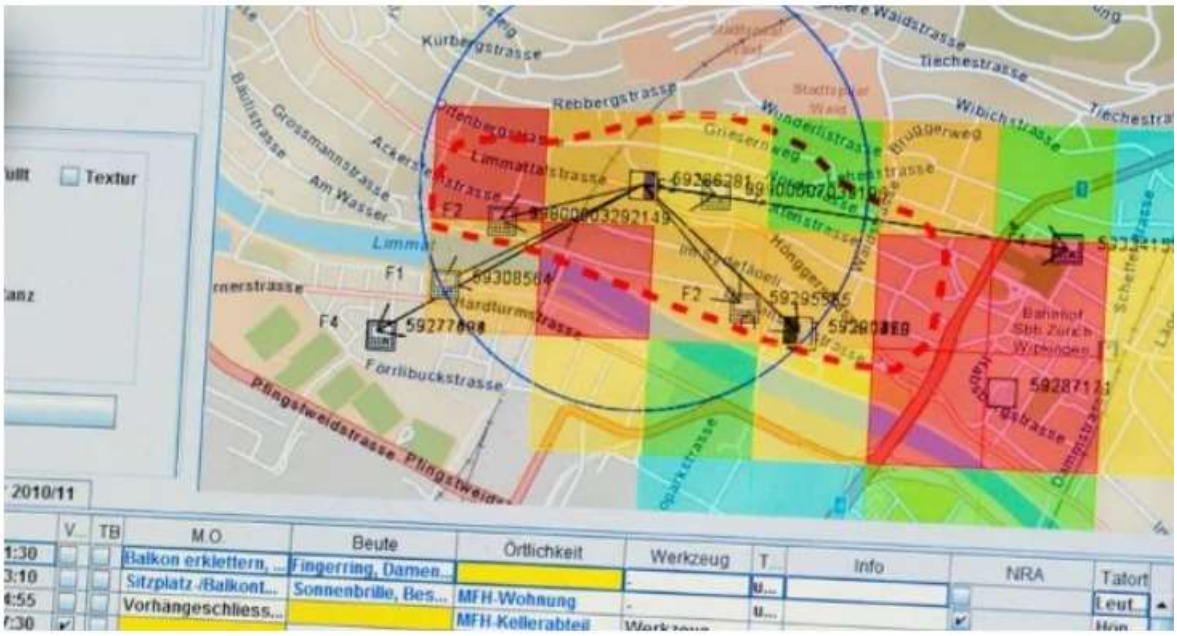


Рис. 1.3. Інтерфейс Precobs.

Модель випадків які повторюються, адаптована Precobs, стверджує, що пограбування з великою ймовірністю повторюються в тих же або близьких місцях через 2, 7 або через 28 днів.

1.3. Інтерактивні карти криміногенності районів міста

Спосіб публікації локальних кримінальних даних, коли вони для зручності відображаються на картах, отримав назву Crime Mapping. Сучасні карти криміногенності Crime Mapping, які акумулюють статистику по правопорушеннях, існують у багатьох країнах з розбивкою по окремих містах. Однак у більшості випадків ці дані відображають картину без прив'язки до унікальних особливостей конкретного району. Для профілактики правопорушень ця інформація повинна бути інтегрована з

геоінформаційними сервісами (ГІС) кожного регіону. Наприклад, маркування потенційно-небезпечних об’єктів (закинуті будівлі, гаражні комплекси, пустирі, лісопаркові зони), а також місця скуплення людей (освітні установи, торгово-розважальні центри) дозволить виявити можливі локації правопорушень.

Crime Mapping – це яскравий приклад аналітики, коли інформація за статистикою правонарушень доповнена множиною неочевидних на перший погляд характеристик місцевості, мітками часу та параметрами самих злочинів. Детальне маркування дозволяє запустити на цих наборах даних алгоритми машинного навчання для кластеризації подій, що відбулися, і прогнозування майбутніх подій, щоб завчасно прийняти попередні заходи.

Інтерактивність криміногенної карти передбачає не тільки постійне оновлення, але й аналітику в режимі онлайн. Для цього її інтегрують із системами міського відеоспостереження. Для прикладу, в Лондоні в 2022 р. було встановлено 640 000 камер, в тому числі 15 000 в метро. Таким чином, середньостатистичний житель мегаполісу попадає в об’єктиви 300 разів за добу. Записи зберігалися протягом 2 тижнів, що допомогло розкрити близько 95 % випадків вбивств, скоєних у місті. Пекінська система відеоспостереження допомогла скоротити кількість автомобільних угонів на 76 %, а загальне число правопорушень – на 38 %.

Подібні системи Big Data з модулями аналітики, в тому числі алгоритмами розпізнавання осіб на базі Machine Learning, дозволяють не тільки розкрити вже здійснені злочини, але попередити правопорушення. Наприклад, на багатьох стадіонах Європи працює біометрична система ідентифікації особистості. Вона виявляє особи, по відношенню до яких є заборона на відвідування місць проведення офіційних спортивних заходів, і успішно функціонує в режимах однофакторної та двофакторної ідентифікації вболівальників.

У якості даних для Crime Mapping представлені не тільки камери міського відеоспостереження, але й мобільні пристрої Інтернету речей (Internet of Things) – дрони. Наприклад, поліція США за допомогою беспілотників відновлює обставини правопорушень, збираються докази з висоти пташиного польоту. Понад 900 американських організацій (поліція, пожежна служба, відділи з надзвичайної ситуації та громадської безпеки та підрозділи дорожнього патрулювання) використовують безпілотники. Зокрема слідчі задіяли дрони для створення трьохвимірних сцен дорожньо-транспортних пригод з багатьма жертвами.

Об’єднуючи результати відеонагляду з геоінформаційними маркерами території, можна побудувати предиктивні моделі майбутніх правопорушень і запобігти їм. На основі великих даних системи Big Data з алгоритмами Machine Learning визначають вірогідність правопорушення з чіткою локалізацією місця та часу.

Більшу частину жителів хвилюють питання, скільки і яких порушень зареєстровано саме в їхньому районі, на їх вулиці, в який час і в які дні спостерігається зростання кількості правопорушень, затриманих зловмисників у кожному окремому інциденті. У багатьох розвинених країнах дані про злочинність можна знайти в інтернеті.

Аналітика великих даних і машинне навчання допомагають у розслідуваннях правопорушень, які пов’язані з незаконним обігом наркотиків. Сьогодні в інтернеті можна купити все що завгодно. У тому числі й наркотики, інформація про які замаскована під текстовий опис цілком легальних товарів. Типові методи автоматизованої обробки тексту на базі алгоритмів Machine Learning не справляються із завданнями ідентифікації забороненого контенту, гарантуючи точність не вище 70 %. Крім того, повідомлення про продаж наркотиків можуть бути розміщені на сторонніх сайтах у коментарях до новин. Таким чином, боротися з цим злом шляхом прямого блокування ресурсу недоцільно.

Система машинного навчання здатна фільтрувати небезпечну інформацію від нейтральної та обґрунтовувати свої висновки. Метод базується на комбінації штучних нейронних мереж і експертних знань лінгвістів та спеціалістів з машинного навчання. Нейромережа шукає певний контент за словами, присвоює їм вагові коефіцієнти і визначає ймовірність того, що конкретний сайт містить заборонену інформацію.

Спочатку досліджується структура контенту, потім за допомогою словника виконується лінгвістичний аналіз вмісту. Далі обчислюється оцінка зв’язку тексту з темою наркотиків. При цьому враховується посилальний характер даних, коли одне повідомлення на веб-сторінці посилається на інші джерела. Завдяки аналізу таких семантичних ланок забезпечується постійна перевірка підозрювальної лексики та поповнення словникової бази. Для оновлення словника назв наркотичних засобів залучаються експерти-наркологи, які знають цей сленг.

Продовжуючи тему виявлення каналів збуту наркотичних речовин, можна зауважити тісну інтеграцію онлайн-ресурсів з офлайн-точками. Це набори цифр, які написані вандалами на стінах будинків, парканах та інших подібних поверхнях уздовж дороги або пішохідних тротуарів. Насправді ці цифри – телефони і акаунти наркоторговців. Вислідити злочинців по ір досить складно, так як вони виходять в мережу через проксі-сервери, інтернет-клуби та інші анонімайзери, а також змінюють адреси, паролі та номери. Додаткову складність у пошуку злочинців несе простота електронних розрахунків між покупцями та продавцями наркотиків через електронні гаманці. Тим не менш, технології великих даних дозволяють збирати і аналізувати цілі ланки, на перший погляд, абсолютно не пов’язаних з подіями. Наприклад, якщо розглядати напис на стіні не тільки як акт вандалізму, але й потенційний канал збуту, можна визначити частоту контактів за цим номером і розслідувати ланку фінансових переводів. Також фіксація правонарушень, які пов’язані з написами, дозволить виявити неблагополучні місця з точки зору міського

планування та прийняти відповідні заходи: покращити місцеве освітлення, демонтувати непотрібну загорожу або закинуту будову. Однак для цього необхідно спочатку визначити дані для аналізу, а потім автоматизувати процеси їх збору та обробки за допомогою інструментів Big Data.

Інтелектуальні системи відеоспостереження, встановлені в аеропортах, залізничних вокзалах та інших місцях перетину державних кордонів, здатні виявляти осіб, які перевозять наркотики. Зокрема, кур'єр старається не допустити перенапруження м'язів, приймаючи специфічні пози. Також до специфічної поведінки відноситься загальмованість рухів, відсутність/мала кількість багажу або камуфлювання за допомогою об'ємних, але легких сумок. Сучасні алгоритми машинного навчання, вбудовані в систему Big Data розпізнавання осіб, здатні ідентифікувати людей з такими ознаками. Таким чином, засоби Data Science допомагають співробітникам поліції виявляти злочинців і проводити експертизу для обліку накопичених даних і самонавчання в режимі онлайн.

ВИСНОВКИ ДО РОЗДІЛУ 1

В першому розділі проаналізовано предметну область, засоби та технології проектування інформаційної системи аналізу правопорушень. З допомогою цієї системи можна буде оцінити рівеь злочинності у певному районі населеного пункту. Показано, що органи влади можуть використовувати дані про злочинність для розробки та оцінки політик безпеки. Це може включати прийняття рішень з питань покращення безпеки в областях міста або вдосконалення кримінального законодавства. Загальний підхід до аналізу та використання даних про злочинність полягає в тому, щоб надавати зрозумілу інформацію та використовувати її для прийняття обґрунтованих рішень з метою покращення безпеки та якості життя громадськості.

РОЗДІЛ 2. ІНФОРМАЦІЙНЕ ЗАБЕЗПЕЧЕННЯ

2.1. Опис і попередній аналіз даних

Цей набір даних відображає зареєстровані випадки злочинів, які сталися в місті Нью-Йорк з 2001 року до 2022 р. Дані отримані з системи CLEAR (аналіз та звітність громадянського правоохоронного органу.

Опис набору даних:

ID – унікальний ідентифікатор для запису;

Case Number – поліцейський номер, унікальний для інциденту;

Date – дата, коли стався інцидент;

Block – відредагована адреса, де стався інцидент, розміщення в тому самому блоці, що й фактична адреса;

IUCR – уніфікований код звітності про злочини штату;

Primary Type – первинний опис коду;

Description – вторинний опис коду, підкатегорія первинного опису;

Location Description – опис місця, де стався інцидент;

Arrest – чи було здійснено арешт;

Domestic – вказує, чи був інцидент пов’язаний із сім’єю, як це визначено законом про домашнє насильство;

Beat – вказує місце, де стався інцидент;

District – вказує округ поліції, де стався інцидент;

Ward – район, де стався інцидент;

Community Area – громадська територія - вказує на територію громади, де стався інцидент;

FBI Code – вказує на класифікацію злочину, як зазначено в національній системі звітності про випадки злочину;

X Coordinate – координата x місця, де стався інцидент;

Y Coordinate – координата Y місця, де стався інцидент;

Year – рік, коли стався інцидент;

Updated On – дата й час останнього оновлення запису;

Latitude – широта місця, де стався інцидент;

Longitude – довгота місця, де стався інцидент;

Location – місце, де стався інцидент, у форматі, який дозволяє створювати карти та виконувати інші географічні операції на цьому порталі даних.

Всього в наборі даних представлено більше сорока типів правопорушень. Однак більша частина з них представлена досить невеликою вибіркою, що робить дані типи правопорушень непридатними для аналізу. Тому для подальшого розгляду було обрано найбільш поширені та небезпечні види правопорушень, число яких за тиждень перевищує 100:

- напади;
- побої;
- викрадення з проникненням;
- причинення збитку;
- замах на правопорушення;
- викрадення транспортних засобів;
- правопорушення, пов’язані з наркотиками;
- пограбування.

Набір даних для аналізу містить невелике число ознак, однак такі ознаки, як час і географічні координати дозволяють більш детально подивитися на те, від чого залежить кількість правопорушень.

2.2. Візуалізація даних

На рис. 2.1 наведені дані про загальну кількість правопрушень. Загальна лінія на графіку являє собою експоненціальне згладжене число злочинів. Як видно таким правопорушенням як напади, побої, викрадення з проникненням, грабежі властива річна сезонність. Також видно, що загальне число випадків злочинів знижується до 2015 року. Однак варто відзначити, що в таких злочинах, як викрадення з проникненням і

викрадення транспортних засобів спостерігаються піки в 2005 та 2011 роках.

Розглянемо календарні залежності злочинності. Нижче наведено дві гістограми для побоїв і нападів, на яких показана залежність числа правопорушень від місяця (рис. 2.2). Видно, що більше правопорушень здійснюється в теплі місяці. Також на кожній гістограмі видно, що в січні відбувається більше правопорушень, ніж в інші зимові місяці.

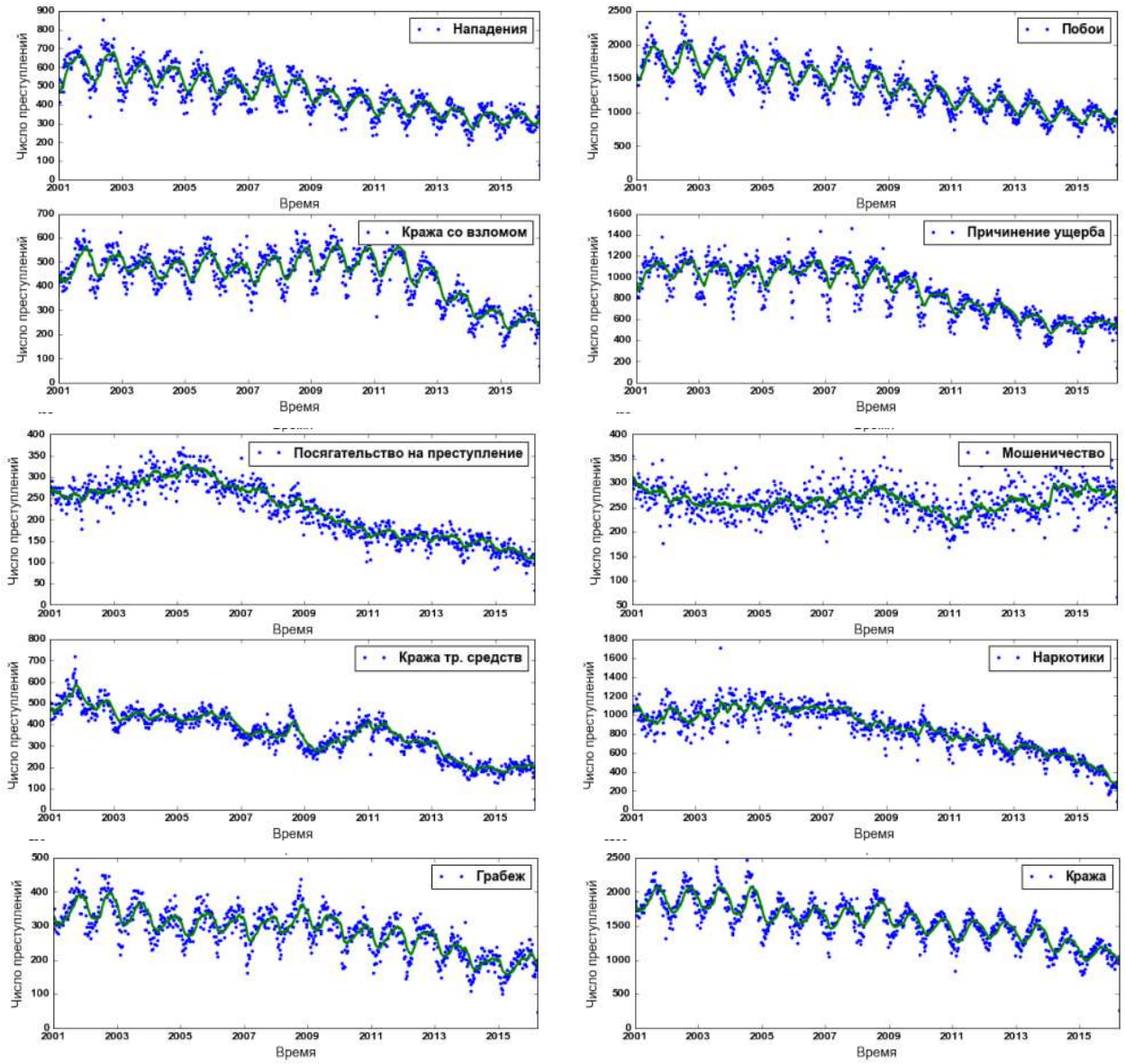


Рис. 2.1: Число злочинів кожного типу.

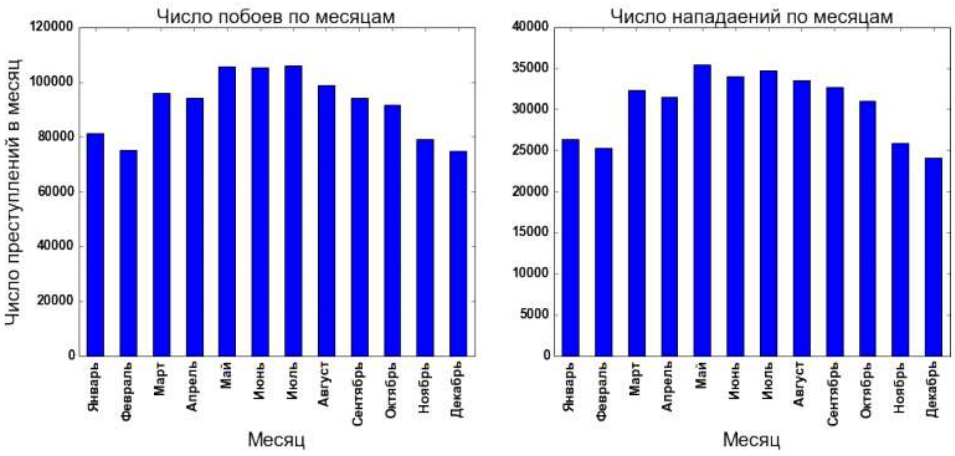


Рис. 2.2. Число злочинів по місяцях.

Крім цього, серйозний вплив має також час протягом доби та номер дня в місяці (рис. 2.3). Теплова карта показує, що набагато більше правопорушень здійснюється в перший день місяця на самому початку доби (0 годин). Також спостерігається пік в південь кожного дня, а також більше злочинів здійснюється ввечері. Цікаво також, що посередині місяця (15 числа) кількість правопорушень зростає.

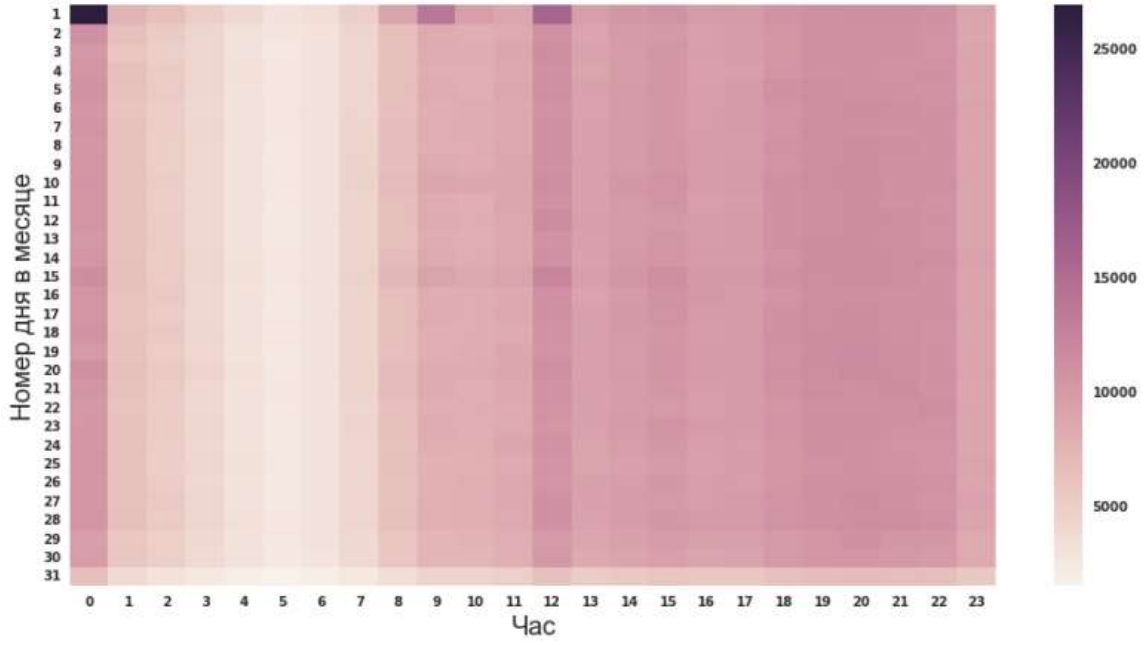


Рис. 2.3. Теплова карта числа злочинів в залежності від часу і дня місяця.

Наступний графік дає уяву про те, як розподіляються правопорушення по днях тижня та по часу (рис. 2.4). Можна зауважити, що більше

правопорушень відбуваються опівдні, ввечері, а також вночі з п’ятниці на суботу та з суботи на неділю. Також видно, що нічних правопорушень в цілому здійснюється менше, а на вихідних цей мінімум зміщений.

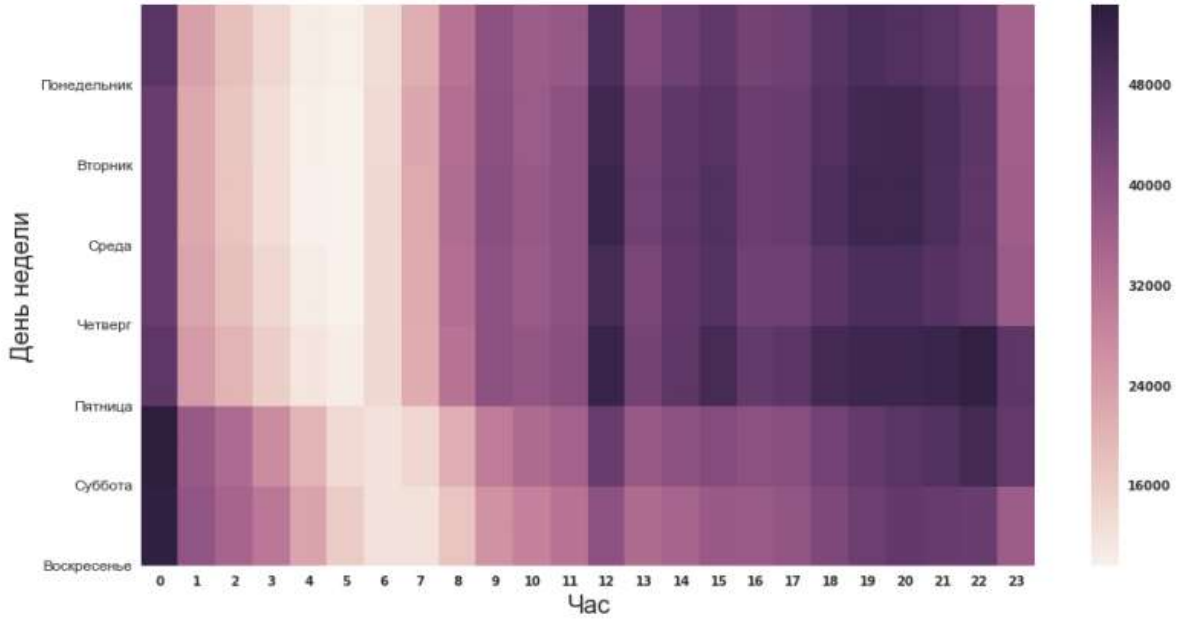


Рис. 2.4: Теплова карта числа злочинів в залежності від часу та дня тижня на місяць.

Розглянемо дані, за якими був побудований графік на рис. 2.4, в розрізі по типах правопорушень, і для кожного типу побудовано гістограми (рис. 2.5). Видно, що побої, спричинення збитку частіше відбуваються у вихідні, напади, викрадення з проникненням, мошенничество, поширення наркотиків і звичайні грабунки, навпаки, частіше відбуваються в будні дні і особливо в п’ятницю. Викрадення транспортних засобів також відбуваються більше в п’ятницю, ніж в інші дні.

Рис. 2.5. Діаграма розкиду для кожного типу злочину по днях тижня.

Найбільше число злочинів здійснюється в теплі місяці. Для перевірки цієї гіпотези було отримано архівні дані про погоду в Нью-йорку і побудовано порівняльний графік (рис. 2.6). Видно, що середня температура і кількість правопорушень, для яких раніше була показана сезонність, пов’язані з тим, що більшість злочинів відбуваються в теплу

погоду. Для кожного злочину був підрахований коефіцієнт кореляції Пірсона.

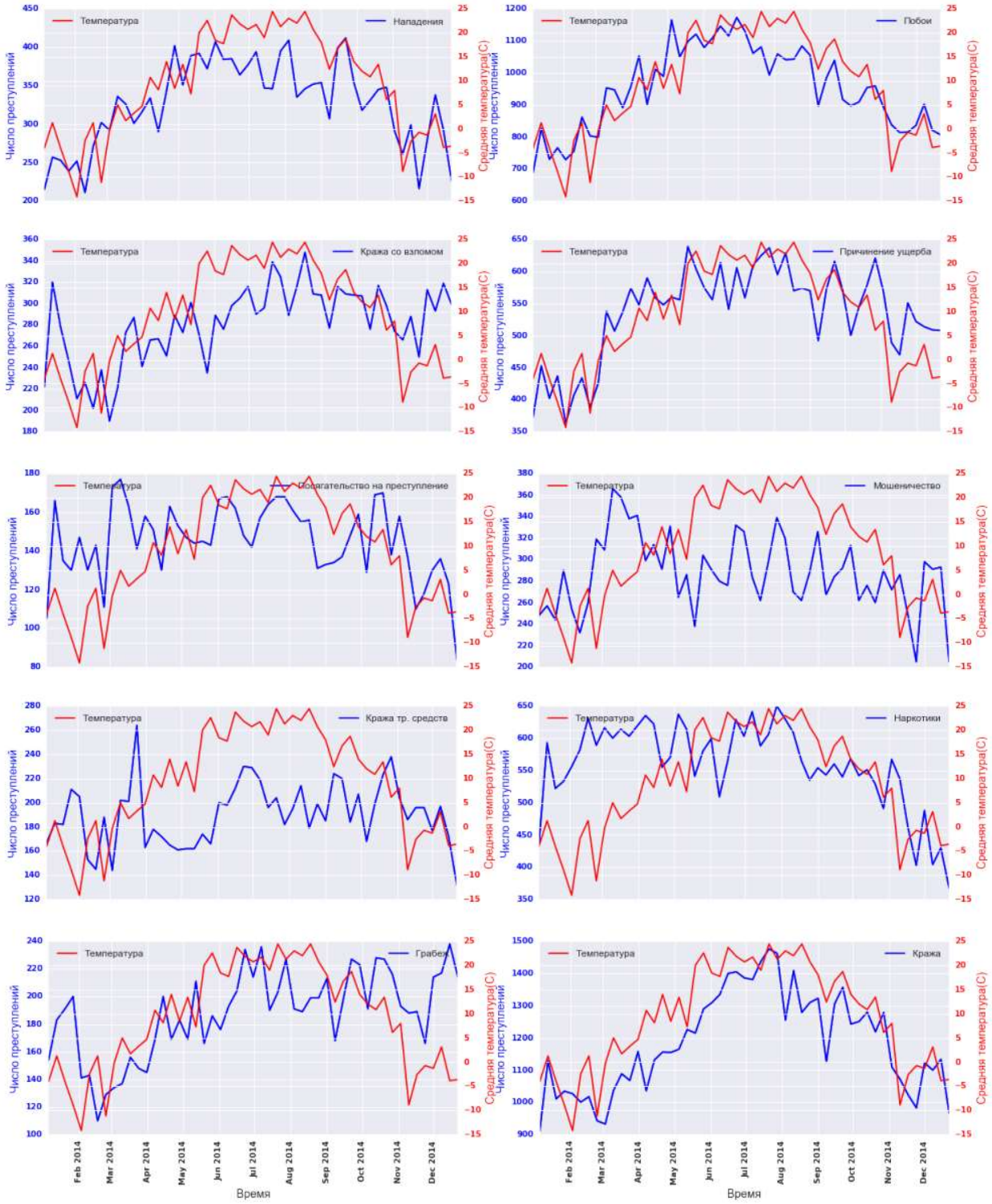


Рис. 2.6. Середня температура та кількість злочинів.

2.3. Географічні температурні карти

Для більш наглядної демонстрації були побудовані географічні температурні карти. Видно, що багато правопорушень відбувається вздовж побережжя, а також у східній і південній частині міста. Також була побудована інтерактивна карта міста, на якій можна переглянути найбільш криміногенні райони та отримати опис злочинів. Кримінальність району може впливати на ціну нерухомості або вибір місця проживання. Тому були побудовані інтерактивні карти, за якими можна оцінити безпеку того чи іншого житлового району міста. Найбільш кримінальні райони знаходяться вздовж побережжя, на сході та півдні міста.

ВИСНОВКИ ДО РОЗДІЛУ 2

Приведено відомості про структуру наобу даних, особливості проектування програмного продукту засобами Python, обробки даних з допомогою бібліотеки Pandas, візуалізації результатів з допомогою бібліотеки Seaborn. Отримані результати аналізу даних про злочинність в Нью-Йорку мають значення для різних сфер діяльності. Аналіз розподілу злочинів за різними категоріями, регіонами та часовими періодами може допомогти місцевим органам визначити та вжити ефективні заходи безпеки для громадськості. Це може включати підвищення поліцейського патрулювання в особливо проблемних районах або впровадження профілактичних програм для зменшення кількості злочинів певних типів. Знання розподілу злочинів за різними регіонами та періодами може допомогти при плануванні ресурсів, таких як поліцейські сили чи екстрені служби. Це дозволяє ефективно розподіляти обмежені ресурси туди, де вони можуть бути найбільш корисними.

РОЗДІЛ 3. МАТЕМАТИЧНЕ ЗАБЕЗПЕЧЕННЯ

3.1. Математична модель прогнозування рівня злочинності

Для подальшого аналізу та побудови моделей машинного навчання були відібрані такі види правопорушень:

- напади;
- побої;
- викрадення із взломом;
- спричинення втрат;
- грабежі;
- злодійство.

Набір даних був перетворений для аналізу, і для кожного виду правопорушення були представлені такими ознаками: число правопорушень даного типу, скоєних у поточний день (цільова змінна); рік; місяць; число місяця; номер дня в році; день тижня; температура повітря; вологість повітря.

Для побудови датасету з попередніми значеннями часового ряду будуть використовуватися зовнішні фактори, такі як день тижня або погодні умови. Тому тільки авторегресійні моделі тут не підходять, і необхідно використовувати алгоритми регресії та авторегресії спільно. Для цього добре підходять моделі на основі алгоритмів машинного навчання, які приймають на вхід матрицю ознак, а на виході дають значення цільової змінної.

Матрицю ознак отримати досить просто, рядок такої матриці буде відображати сукупність описаних ознак на один день (рік, місяць, день тижня), а відповіддю на один такий набір ознак буде значення числа правопорушень, що відбулися в цей день. Таким чином, весь набір даних буде перетворений в матрицю ознак X, яка матиме такий вид:

$$X = \begin{pmatrix} x_{11} & x_{12} & x_{13} & \dots & x_{1n} \\ x_{21} & x_{22} & x_{23} & \dots & x_{2n} \\ x_{31} & x_{32} & x_{33} & \dots & x_{3n} \\ \dots & \dots & \dots & \dots & \dots \\ x_{n1} & x_{n2} & x_{n3} & \dots & x_{nn} \end{pmatrix} \quad (3.1)$$

та вектору відповідей *Y*:

$$Y = \left(y_1, y_2, y_3, \dots, y_n \right)^T . \quad (3.2)$$

Для побудови моделі машинного навчання використано мову програмування Python та пакет Scikit-Learn. Так як прогнози були побудовані для 7 типів правопорушень, то, цілком можливо, що одна модель не зможе описати кожен із типів правопорушень, тому було обрані 4 алгоритми:

- регресія на базі випадкового лісу;
- регресія на базі вирішальних дерев;
- регресія на базі SVM.

Раніше булоотримано матрицю ознак X і вектор відповідей Y для кожного типу правопорушень. Рядки матриці X називаються об’єктами, а відповідні елементи вектора Y відповідями на цих об’єктах. Кожен із представлених алгоритмів намагається встановити невідому залежність у X → Y саме методом побудови наближення, а алгоритми відрізняються від іншого.

Вирішальне дерево являє собою в дереві листя, яке має значення апроксимуючої функції, а вузли представляють собою умови переходу по ребрах. На кожному етапі алгоритму по кожній ознаці будується роздільна площина, і для кожної частини вибору, яка розділена площиною, передбачається середнє значення відповідей об’єктів, що потрапили в цю частину простору, і на основі цього обчислюється середньоквадратична помилка для кожного з напівпростору. У підсумку вибирається площина, яка розбиває простір так, що сумарна середньоквадратична помилка

мінімальна. Настроюваними параметрами моделі є глибина дерева, кількість вимірів, мінімальна кількість об’єктів у листі.

Алгоритм регресії на базі випадкових лісів являє собою композицію множини вирішальних дерев, які представляють собою сильно висічені, і побудовані лише на підмножині визнаних дерев. Справа в тому, що в окремості цих дерев немає ніякої цінності, оскільки вони роблять досить неточні прогнози, однак якщо взяти велике число таких дерев і усереднити всі їх покази, то прогнози будуть достатньо точні. Настроюваними параметрами в даній моделі є число дерев, кількість визначень для навчання одного дерева, а також параметри, які використовуються при навчанні звичайних дерев.

Для пошуку оптимальної моделі частина даних була відкладена для втрати й оцінки самого прогнозу. На решті частини даних методом перекрестної перевірки за набором параметрів підбирався кращий алгоритм і його параметри. Результат кожного алгоритму з кожним набором параметрів оцінювався за метрикою *MAPE* (середня абсолютна відсоткова помилка), яка дозволяє визначити середню помилку прогнозу:

$$MAPE = \sum_{i=1}^{k} \frac{\left| y_i - \hat{y}_i \right|}{y_i} \cdot 100\ \%. \qquad (3.3)$$

Часовим рядом називається сукупність вимірів y_1 , y_2, . . . , y_n, . . ., де y_n - деякі величини через визначені проміжки часу. Задача передбачення часового ряду записується таким чином:

$$\hat{y}_{t+d} = f_t\left(y_1, y_2, ..., y_n\right) \qquad (3.4)$$

де $\hat{y}_{t+d}$ - передбачене значення шуканої величини в якийсь момент в майбутньому, α - параметр моделі, z - інші фактори, що впливають на величину, яка вимірюється. Для цього потрібно знайти якусь функцію f_t, яка на основі попередніх значень часового та інших факторів буде передбачати наступні її значення.

Основні явища, які можуть спостерігатися у часових рядах:

- тренди;
- сезонності;
- зміна моделі.

Роглянемо найбільш поширені методи прогнозування часових рядів. Часто на прогнозовану величину впливають багато сторонніх факторів. Метою регресійного аналізу для прогнозування часових рядів є побудова деякої функціональної залежності між прогнозованою величиною та цими факторами. Самий простий варіант регресійної моделі - це лінійна регресія. Є якась змінна x, потрібно передбачити значення часового ряду y_t на основі значень цієї змінної:

$$y_t = \alpha_0 + \alpha_1 x_t + e_t, \tag{3.5}$$

де α_0 та α_1 - коефіцієнти регресії; e_t - похибка моделі.

Коли існує кілька факторів, що впливають на модель, то використовується множинна регресія:

$$y_t = \alpha_0 + \sum_{i=1}^{n} \alpha_i x_{it} + e_t . \tag{3.6}$$

Якщо значення часового ряду мають нелінійну залежність від факторів, то застосовують нелінійну регресію:

$$y_t = F\left(x_t,\ \alpha\right) + e_t, \tag{3.7}$$

де F - функція, яка, добре описує взаємозв'язок між значеннями факторів і значеннями часового ряду, α, x та t - вектори коефіцієнтів і факторів.

Коефіцієнти моделі підбираються на основі наявних даних шляхом мінімізації помилок, наприклад, методом найменших квадратів. На практиці часто буває важко отримати значення зовнішніх факторів у той же момент часу, в якому зберігається прогноз, що є великим недоліком регресійних моделей.

3.2. Прогнозування значень часового ряду злочинності

Розглянемо наступну формулу для прогнозування значень часового ряду:

$$y_t + 1 = \sum_{i=0}^{n} \alpha_i f y_{t-1}, \tag{3.8}$$

тобто для передбачення наступного значення використовується n попередніх значень ряду. Звідси можна отримати наступну матрицю ознак і вектор відповідей для кожного набору ознак:

$$X = \begin{pmatrix} y_{t-1} & y_{t-2} & y_{t-3} & \cdots & y_{t-n} \\ y_{t-2} & y_{t-3} & y_{t-4} & \cdots & y_{t-n-1} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ y_{n-1} & y_{n-2} & y_{n-3} & \cdots & y_0 \end{pmatrix}, \quad Y = \begin{pmatrix} y_t \\ y_{t-1} \\ \cdots \\ y_n \end{pmatrix} \tag{3.9}$$

У такій постановці завдання може вирішуватися самими різними методами машинного навчання, оскільки вхідними даними для них є матриці ознак і відповідей на них. Наприклад, можна застосувати методи побудови дерев класифікації та регресії.

ВИСНОВКИ ДО РОЗДІЛУ 3

Математичне моделювання правопорушень є потужним інструментом для аналізу та передбачення динаміки злочинності. Засновані на статистичних алгоритмах та обробці великих об’ємів даних, такі моделі можуть надавати важливі уявлення та сприяти прийняттю обгрунтованих рішень у сфері правопорядку. Математичне моделювання правопорушень в контексті аналізу даних про злочинність надає можливість розуміння факторів, що впливають на злочинність в різних регіонах та часових періодах. Модель виявляє взаємозв’язки між різними змінними, такими як часові та просторові фактори, характеристики злочинів та демографічні дані.

На основі результатів моделювання можна здійснювати прогнози та розробляти стратегії кримінального прогнозування. Враховуючи динаміку змін у розподілі злочинів, правоохоронні органи та органи влади можуть вдосконалювати плани боротьби з правопорушеннями та ефективно розподіляти ресурси для забезпечення безпеки громадськості. Математичне моделювання дозволяє враховувати багатофакторність та динамічний характер злочинності, надаючи аналітикам та рішенням у сфері безпеки цінні інструменти для стратегічного управління та проактивного реагування на зміни в кримінальному середовищі.

РОЗДІЛ 4. ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ

4.1. Проведення попереднього аналізу даних

При проектуванні інформаційної системи на попередньому етапі потрібно провести попередній аналіз наявних даних. Щоб це провести, імпортують необхідні бібліотеки та їх функціонал для роботи з даними та візуалізації на мові Python. Обрані бібліотеки широко використовуються у сферах аналізу даних та візуалізації, що дозволяє ефективно працювати з різноманітними завданнями та типами даних. NumPy - це бібліотека для наукових обчислень в Python, Pandas - для обробки та аналізу даних. Бібліотеки Matplotlib та Seaborn - для створення графіків та візуалізації даних, для створення статистичних графіків. Бібліотека Folium використовується для створення веб-карт з використанням Leaflet.js. Модуль Folium.plugins містить додаткові плагіни для Folium, такі як HeatMap, який використовується для візуалізації теплових карт. Модуль Folium.features містить різні можливості для взаємодії з об'єктами на карті. ClickForMarker - це одна з таких можливостей, яка дозволяє додавати маркери при натисканні на карту. Branca.element - модуль, який містить елементи для створення веб-карт. Template і MacroElement - це частини цього модуля.

```
df = pd.read_csv('ny_clean_all.csv')
```

Для отримання даних в Pandas використовують функцію read_csv для зчитування даних з CSV-файлу та їх завантаження в дата фрейм. Файл імпортують з назвою ny_clean_all.csv. df` - ім'я, яке вибрали для зберігання отриманих даних у вигляді датафрейму. Датафрейм - основна структура даних, яка надає зручний та ефективний спосіб роботи з табличними даними у Python. Можна використовувати цей датафрейм в подальшому для здійснення операцій з обробки, аналізу та візуалізації даних.

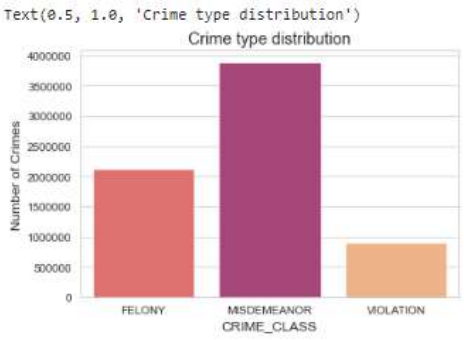
df.info() - це метод об'єкта датафрейм, який надає інформацію про його вміст. Виклик цього методу надасть можливість отримати загальну

структуру та характеристик даних. Його використовують для отримання такої інформації:

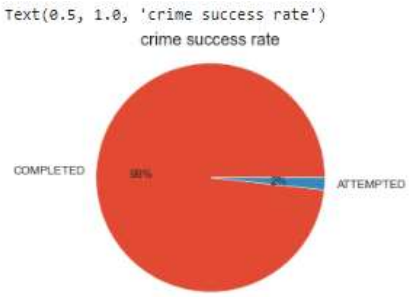
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6882148 entries, 0 to 6882147
Data columns (total 23 columns):
 #   Column             Dtype
---  ------             -----
 0   CMPLNT_NUM         int64
 1   year               int64
 2   month              int64
 3   day                int64
 4   weekday            object
 5   hour               int64
 6   Latitude           float64
 7   Longitude          float64
 8   CRM_ATPT_CPTD_CD   object
 9   OFNS_DESC          object
 10  ADDR_PCT_CD        float64
 11  CRIME_CLASS        object
 12  BORO_NM            object
 13  PREM_TYP_DESC      object
 14  IN_PARK            int64
 15  IN_PUBLIC_HOUSING  int64
 16  IN_STATION         int64
 17  SUSP_AGE_GROUP     object
 18  SUSP_RACE          object
 19  SUSP_SEX           object
 20  VIC_AGE_GROUP      object
 21  VIC_RACE           object
 22  VIC_SEX            object
dtypes: float64(3), int64(8), object(12)
memory usage: 1.2+ GB
```

Видно, що в цей дата фрейм містить 1000 рядків та 5 стовпців. Біля кожного стовпця видно тип даних. Далі використовують бібліотеки Pandas та Seaborn для аналізу та візуалізації кількості злочинів за категоріями. Для цього створюють новий датафрейм з назвою df2, який містить інформацію про кількість злочинів за категоріями. Групування виконується за стовпцем CRIME_CLASS, для кожної категорії підраховується кількість злочинів за допомогою методу count(). Визначаються параметри для графіку. sns.set_style('whitegrid') встановлює стиль графіку, palette визначає палітру кольорів для графіку з використанням Seaborn. data містить кількість злочинів для кожної категорії, rank визначає порядок категорій за кількістю злочинів. Створюють стовпчасту діаграма barplot. Кожна категорія розташована на осі X, кількість злочинів - по осі Y. Кольори беруться з палітри з використанням методу palette, їх порядок визначається

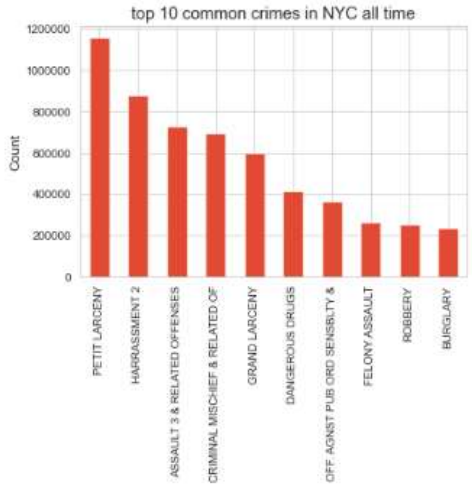
параметром rank. Створюють стовпчасту діаграму, яка візуалізує розподіл кількості злочинів за категоріями (CRIME_CLASS):



Після цього використовують бібліотеку Matplotlib для створення кругової діаграми, яка візуалізує відсотковий розподіл успішних та неуспішних випадків злочинів. Використовується метод pie для створення кругової діаграми. Аргумент labels=df['CRM_ATPT_CPTD_CD'].unique() вказує мітки для різних частин кругової діаграми. Це будуть значення зі стовпця CRM_ATPT_CPTD_CD. Аргумент autopct='%.0f%% додає відсоткові значення на кожен сегмент діаграми. Тобто було визначено та візуалізовано розподіл успішних та неуспішних випадків злочинів у формі кругової діаграми:

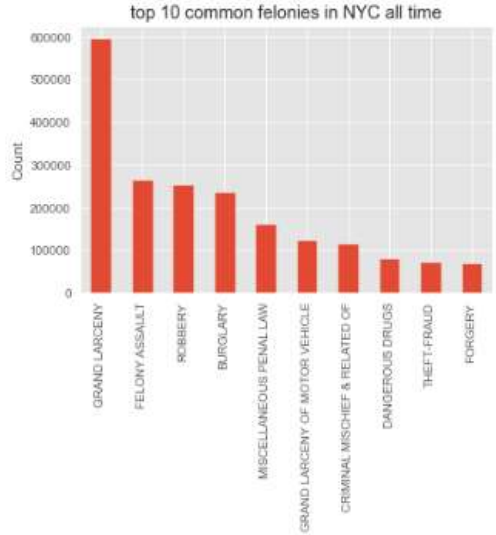


Далі створюють стовпчасту діаграму, на якій буде візуалізовано кількість 10 найпоширеніших видів злочинів в Нью-Йорку за всі часи. Для цього використано метод value_counts() для підрахунку кількості злочинів для кожного унікального опису (OFNS_DESC). Потім з об’єкта Series вибираються перші 10 значень з використанням [:10]. Створюють стовпчасту діаграму методом plot.bar():




```
df[df['CRIME_CLASS']=='FELONY']['OFNS_DESC'].value_counts()[:10].plot.bar()
plt.ylabel('Count')
plt.title('top 10 common felonies in NYC all time')
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```

Створюють стовпчасту діаграму, яка фільтрує дані, використано фільтр для обрання тільки тих записів, де CRIME_CLASS рівний FELONY. Після цього застосовується метод value_counts() для підрахунку кількості кожного унікального опису злочину для вибраних записів.

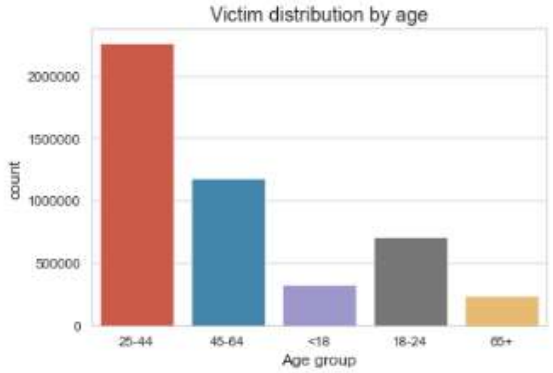



```
x = df[(df['VIC_AGE_GROUP'] !='UNKNOWN')]
sns.countplot(x=x['VIC_AGE_GROUP'])
plt.ticklabel_format(style='plain', axis='y')
plt.xlabel('Age group')
plt.ylabel('count')
plt.title('Victim distribution by age')
```

Далі створюють стовпчасту діаграму, яка візуалізує розподіл віку жертв злочинів в Нью-Йорку. Тут використовується метод countplot для створення стовпчастої діаграми кількості жертв за групами віку.

Аргументx=x['VIC_AGE_GROUP']` відповідає за те, які дані використовувати.

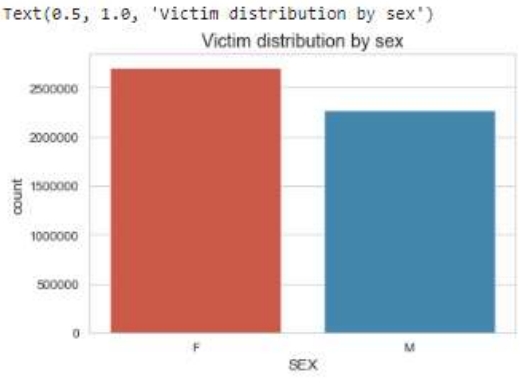
```
Text(0.5, 1.0, 'Victim distribution by age')
```




```
x = df[(df['VIC_SEX']=='M') | (df['VIC_SEX']=='F')]
sns.countplot(x=x['VIC_SEX'])
plt.ticklabel_format(style='plain', axis='y')
plt.xlabel('SEX')
plt.ylabel('count')
plt.title('Victim distribution by sex')
```

На наступному етапі роботи створюють стовпчасту діаграму, яка візуалізує розподіл статі жертв злочинів в Нью-Йорку. З допомогою методу countplot бібліотеки Seaborn буде створено стовпчасту діаграму кількості жертв за статтею:

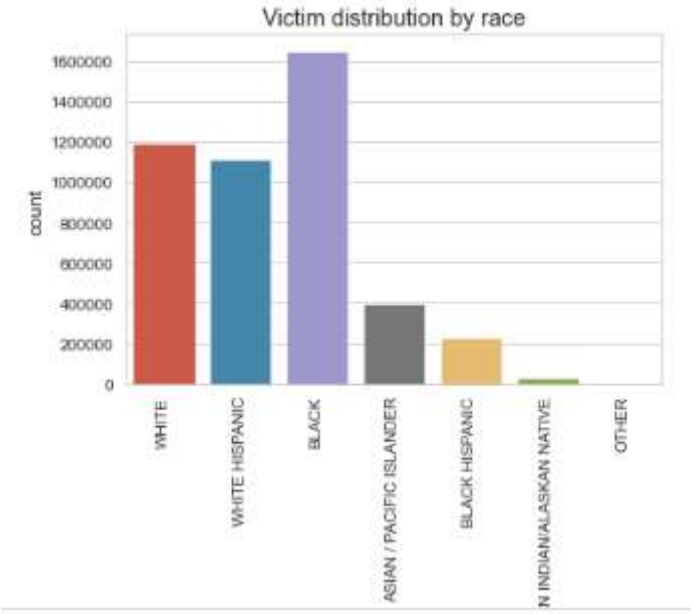
```
Text(0.5, 1.0, 'Victim distribution by sex')
```




```
x = df[(df['VIC_RACE']!='UNKNOWN')]
sns.countplot(x=x['VIC_RACE'])
plt.ticklabel_format(style='plain', axis='y')
plt.xlabel('RACE')
plt.ylabel('count')
plt.xticks(rotation=90)
plt.title('Victim distribution by race')
```

Далі створюють стовпчасту діаграму, на якій буде візуалізовано розподіл раси жертв злочинів в Нью-Йорку. Метод countplot використано для створення стовпчастої діаграми кількості жертв за расою:

```
Text(0.5, 1.0, 'Victim distribution by race')
```




```
colors = {'felony': '#ff0e0a', 'misdemeanor':'#ff8133', 'violation':
'#ffed47'}
```

Далі створюють словник кольорів для різних видів злочинів у датасеті. У цьому словнику felony: #ff0e0a визначає колір для злочинів, класифікованих як felony. misdemeanor: #ff8133 визначає колір для злочинів, класифікованих як misdemeanor. violation: #ffed47 визначає колір для злочинів, класифікованих як violation. Ці колірні коди виглядають як шістнадцяткові представлення кольорів Можна використати цей словник для присвоєння кольорів різним категоріям злочинів під час візуалізації даних для кращого розрізнення.

```
def colorByCrime(crime):
    if crime == 'FELONY':
        return colors['felony']
    elif crime == 'MISDEMEANOR':
        return colors['misdemeanor']
    else :
        return colors['violation']
```

Функція colorByCrime приймає тип злочину як вхідний параметр і повертає відповідний колір на основі заданого словника colors. Це може бути корисно при визначенні кольорів для різних видів злочинів при використанні цих даних у візуалізаціях чи графіках. Можна використати цю функцію для присвоєння кольорів конкретним видам злочинів. Ця функція використовує ваш словник colors, щоб повернути відповідний колір для виду злочину, який передається в якості параметра.

```
def generateBaseMap(default_location=[40.704467, -73.892246],
default_zoom_start=13,min_zoom=11,max_zoom=15,):
    base_map = folium.Map(location=default_location, control_scale=True,
zoom_start=default_zoom_start)
    base_map.add_child(ClickForMarker())
    return base_map
```

Функція generateBaseMap використовує бібліотеку Folium для створення базової карти, яка може відображати різні шари. В цій функції folium.Map створює об’єкт карти Folium з заданою початковою локацією та рівнем масштабування. folium.features.ClickForMarker() додає можливість клікати на карті для встановлення маркера. Можна її використовувати для отримання базової карти та подальшого додавання до неї різних елементів (шарів, маркерів, теплових карт).

```
def crimeByDate(df, base_map, year, month=0, day=0):
    '''
        returns crimes for a specific date YYYY,MM,DD
        returns crimes for a specific month YYYY,MM
        returns crimes for a specific year YYYY
    '''
    assert year, 'please enter at least a year'
    if (month & day):
        map_df = df[(df['year']== year) & (df['month'] == month) &
(df['day']== day)]
    elif month:
        map_df = df[(df['year']== year) & (df['month'] == month)]
    else :
        map_df = df[(df['year']== year)]

    for index, row in map_df.iterrows():
        color = colorByCrime(row['CRIME_CLASS'])
        folium.CircleMarker([row['Latitude'], row['Longitude']],
                            radius = 3,
                            popup = row['OFNS_DESC'],
                            color = color,
                           ).add_to(base_map)
```

Функція crimeByDate отримує датафрейм df, базову карту base_map та фільтрує злочини відповідно до заданих параметрів дати. Вона додає маркери для кожного злочину на карту з додатковою інформацією та кольором в залежності від класу злочину. Ця функція фільтрує датафрейм злочинів відповідно до заданих дати, місяця та року. Для кожного рядка у фільтрованому датафреймі додає маркер на базову карту Folium. Розмір та колір маркера залежать від класу злочину (CRIME_CLASS), а інформація про злочин відображається при наведенні на маркер. Для використання цієї функції слід спочатку створити базову карту за допомогою функції generateBaseMap. Це створить HTML-файл із маркерами для злочинів, які

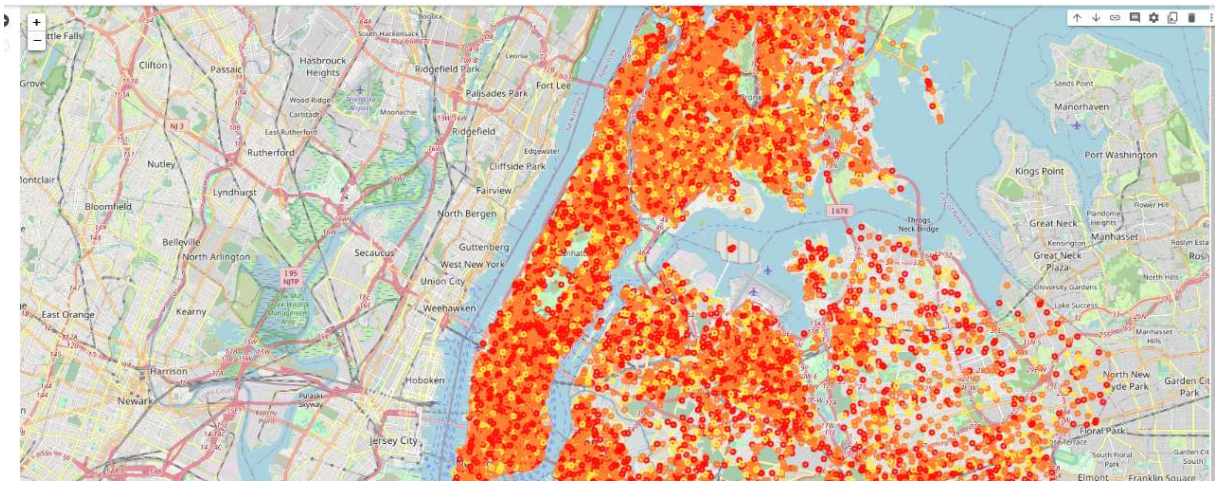
відповідають вказаній даті, який можна відкрити у веб-браузері для перегляду.

```
def heatmapByDate(df, base_map, year, month=0, day=0):
    assert year, 'please enter at least a year'
    if (month & day):
        map_df = df[(df['year']== year) & (df['month'] == month) &
(df['day']== day)]
    elif month:
        map_df = df[(df['year']== year) & (df['month'] == month)]
    else :
        map_df = df[(df['year']== year)]
    dfmatrix = map_df[['Latitude', 'Longitude']].values
    base_map.add_child(plugins.HeatMap(dfmatrix, radius=15))
```

Функція heatmapByDate призначена для створення теплової карти на базовій карті Folium за допомогою даних про злочини. Вона використовує бібліотеку Folium та її розширення HeatMap для візуалізації гарячих точок злочинів. Ця функція фільтрує датафрейм злочинів відповідно до заданих дати, місяця та року. Створює матрицю з координатами широти та довготи для використання в тепловій карті. Додає до базової карти Folium об’єкт HeatMap, який відображає гарячі точки (кластери) на основі матриці координат. Можна викликати цю функцію, передавши необхідні параметри та базову карту, створену за допомогою функції generateBaseMap.

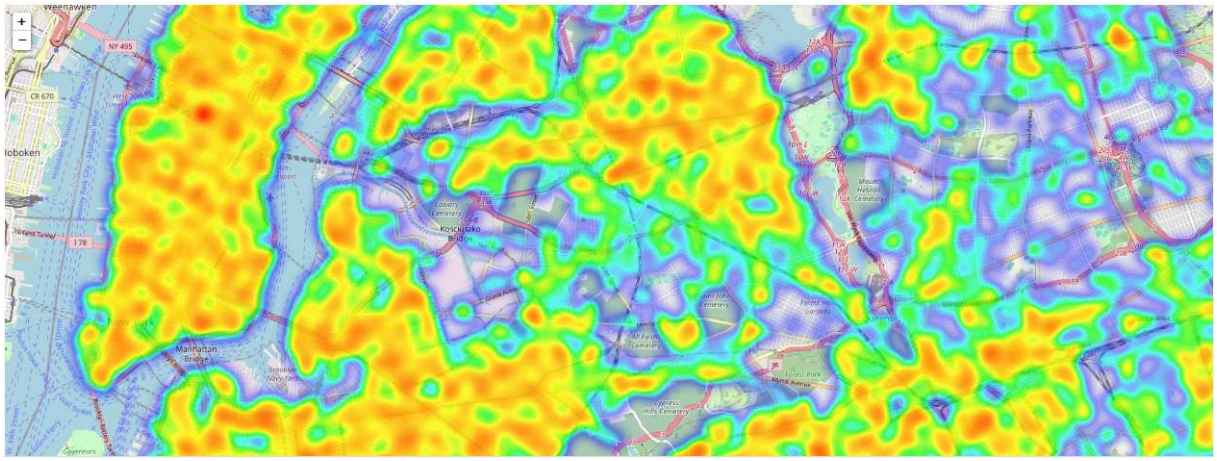
```
base_map = generateBaseMap()
crimeByDate(df, base_map, 2018,4,2)
macro = MacroElement()
macro._template = Template(createLegend())
base_map.get_root().add_child(macro)
base_map
```

У цьому фрагменті коду використана функція generateBaseMap для створення базової карти, потім викликають функцію crimeByDate для відображення розподілу злочинів за типом для конкретної дати. Далі створюють елемент MacroElement, додають до нього шаблон легенди за допомогою функції createLegend та додають цей елемент до кореневого елемента базової карти. Цей код дозволяє відобразити розподіл злочинів за типом на карті для певної дати та додати легенду для пояснення кольорів, які використовуються для позначення різних типів злочинів. Результат:




```
new_map = generateBaseMap()
heatmapByDate(df, new_map, 2018,4,2)
new_map
```

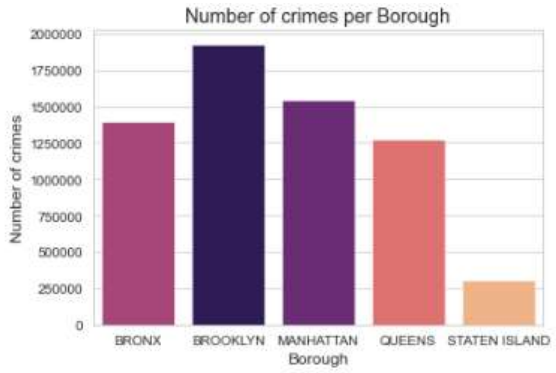
Тут відображено, як створити теплову карту для злочинів на певну дату. Для цього використано функцію generateBaseMap для створення базової карти та функцію heatmapByDate для відображення теплової карти за певну дату. Цей код створить нову базову карту та накладе на неї теплову карту, яка відображає гарячі точки для злочинів на певну дату:




```
df2 =
df[df['BORO_NM']!='UNKNOWN'].groupby(['BORO_NM'])['CMPLNT_NUM'].count().res
et_index()
data =
df[df['BORO_NM']!='UNKNOWN'].groupby(['BORO_NM'])['CMPLNT_NUM'].size()
palette = sns.color_palette("magma",5)
rank = data.argsort().argsort()
g=sns.barplot(x='BORO_NM',y='CMPLNT_NUM',data=df2,palette=np.array(palette[
::-1])[rank]);
plt.xlabel('Borough')
plt.ylabel('Number of crimes');
plt.ticklabel_format(style='plain', axis='y')
plt.title("Number of crimes per Borough");
```

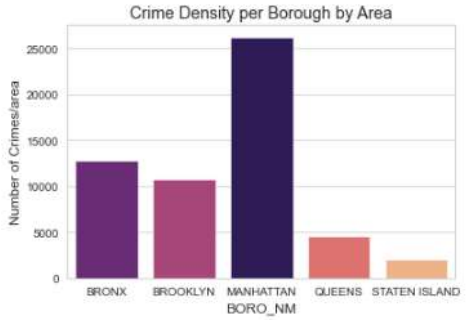
Створюють стовпчасту діаграму, яка показує кількість злочинів у кожному з районів міста Нью-Йорк. Використано бібліотеку Seaborn для

створення графіку, використовуючи дані з фрейму даних df, палітру кольорів magma для візуалізації, де кожний стовпець на графіку відповідає окремому району:



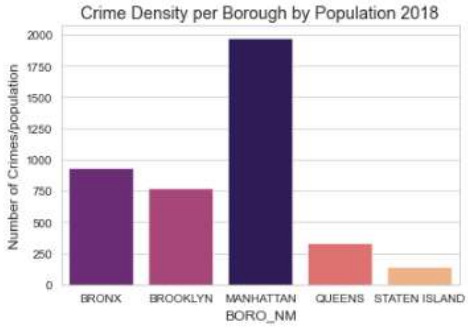

```
borough_area = {'BROOKLYN':179.7, 'STATEN ISLAND':148.9, 'BRONX':109.3,
'QUEENS':281.5, 'MANHATTAN':58.8}
df2 =
df[df['BORO_NM']!='UNKNOWN'].groupby(['BORO_NM'])['CMPLNT_NUM'].count().res
et_index()
df2['Area'] = df2.apply(transform, val_dict=borough_area, column='BORO_NM',
axis=1);
df2['CrimeDensityArea'] = df2.CMPLNT_NUM / df2.Area
df2.head()
data = df2['CrimeDensityArea']
palette = sns.color_palette("magma",5)
rank = data.argsort().argsort()
sns.set_style('whitegrid');
g=sns.barplot(x='BORO_NM',y='CrimeDensityArea',data=df2,palette=np.array(pa
lette[::-1])[rank]);
plt.ylabel('Number of Crimes/area');
plt.title("Crime Density per Borough by Area");
```

Цей код призначений для створення стовпчастої діаграми, яка відображає густину злочинності в кожному з районів міста Нью-Йорк в залежності від площі району. Тут визначають площу для кожного району borough_area та фільтрують дані. Додають інформацію про площу для кожного району до фрейму даних df2 за допомогою функції transform. Розраховують густину злочинності (кількість злочинів на одиницю площі) за допомогою стовпця CrimeDensityArea. Використано бібліотеку Seaborn для створення стовпчастої діаграми, де кожен стовпець відповідає окремому району, а висота стовпця визначає густину злочинності. Візуалізація надає інформацію про те, як різні райони міста Нью-Йорк відрізняються за густиною злочинності враховуючи їхню площу:




```
borough_pop_18 = {'BROOKLYN':2582830, 'STATEN ISLAND':476179,
'BRONX':1432130, 'QUEENS':2278910, 'MANHATTAN':1628700}
df3 = df[(df['BORO_NM']!='UNKNOWN') &
(df['year']==2018)].groupby(['BORO_NM'])['CMPLNT_NUM'].count().reset_index(
)
df3['Population18'] = df3.apply(transform, val_dict=borough_area,
column='BORO_NM', axis=1);
df3['CrimeDensityPop'] = df3.CMPLNT_NUM / df3.Population18
data = df3['CrimeDensityPop']
palette = sns.color_palette("magma",5)
rank = data.argsort().argsort()
sns.set_style('whitegrid');
g=sns.barplot(x='BORO_NM',y='CrimeDensityPop',data=df3,palette=np.array(pal
ette[::-1])[rank]);
plt.ylabel('Number of Crimes/population');
```

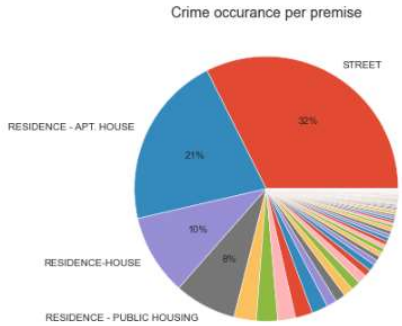
Цей код відповідає за створення стовпчастої діаграми, яка відображає густину злочинності в кожному з районів міста Нью-Йорк в залежності від населення району у 2022 році. Для цього визначають населення для кожного району у 2022 році borough_pop_22. Додають інформацію про населення для кожного району у 2022 році до фрейму даних df3 за допомогою функції transform, розраховують густину злочинності за допомогою стовпця CrimeDensityPop:




```
pct_cutoff=5
fig= plt.figure(figsize=(15,6))
the predefined cutoff value
def my_autopct(pct):
    return ('%1.0f%%' % pct) if pct > pct_cutoff else ''
df_temp=df['PREM_TYP_DESC'].value_counts(normalize=True).round(8)
labels = [n if v > pct_cutoff/100 else ''
          for n, v in zip(df_temp.index, df_temp)]
plt.pie(df_temp, labels=labels, autopct=my_autopct, shadow=False)
```

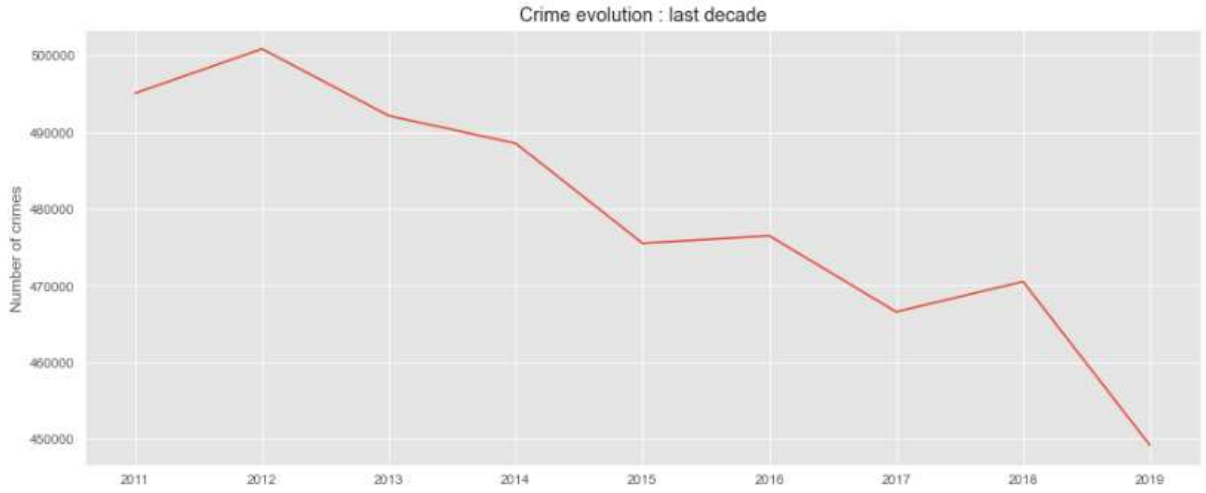
```
plt.title('Crime occurance per premise')
plt.show()
```

Створюють кругову діаграму для відображення частоти виникнення злочинів відносно місць вчинення у вигляді відсотків. Використовують метод plt.pie для створення кругової діаграми з відсотковими значеннями, функцію my_autopct, щоб визначити, коли виводити відсотки, а коли ні, залежно від значення параметру pct_cutoff:



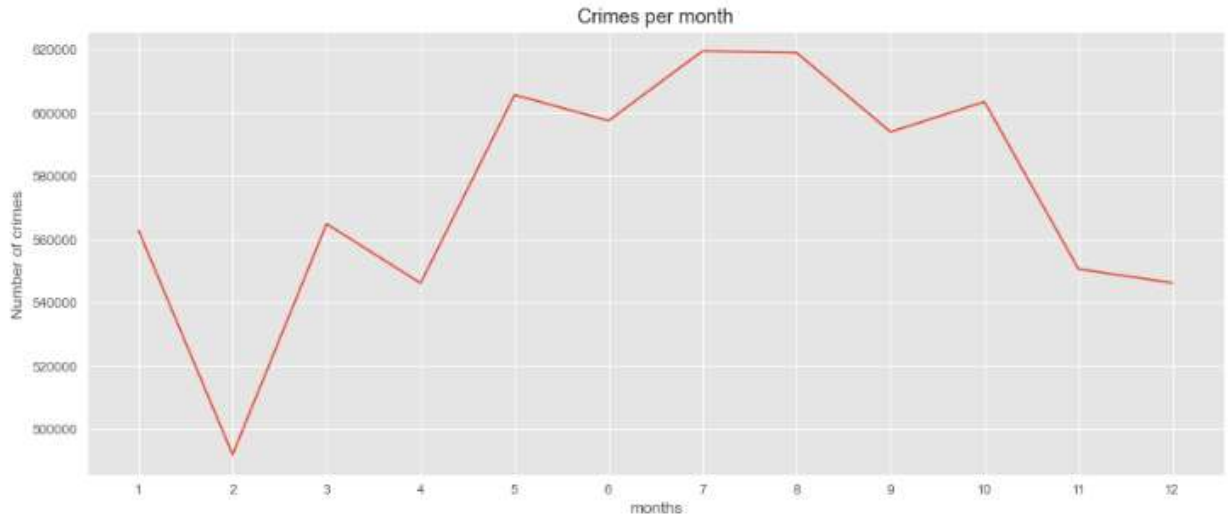

```
fig= plt.figure(figsize=(15,6))
temp_df = df[df["year"]>2010]
temp_df.groupby('year').count()["CMPLNT_NUM"].plot(kind='line')
plt.xlabel('')
plt.ylabel('Number of crimes')
plt.title('Crime evolution : last decade')
```

Створюють лінійну діаграму для відображення розподілу злочинів в залежності від року. Тут використано temp_df для фільтрації даних і відсіювання злочинів, які відбулися до 2010 року. Після цього групують дані за роками, використовуючи метод groupby('year').count()["CMPLNT_NUM"]. Метод plot(kind='line') використано, щоб створити лінійний графік, де по горизонталі відкладено роки, а по вертикалі - кількість злочинів. Цей графік ілюструє еволюцію кількості злочинів за останні десять років:




```
fig= plt.figure(figsize=(15,6))
df.groupby('month').count()["CMPLNT_NUM"].plot(kind='line')
plt.xlabel('months')
plt.xticks([x for x in range(1,13)])
plt.ylabel('Number of crimes')
plt.title('Crimes per month')
```

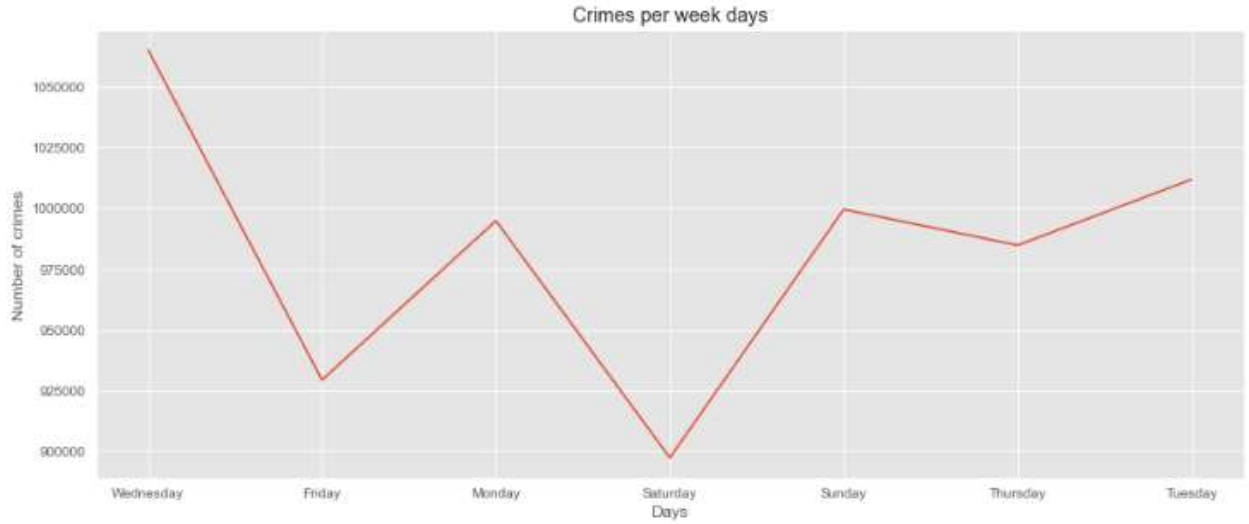
Тут будують лінійну діаграму для відображення розподілу злочинів в залежності від місяця. Використано метод groupby('month').count()["CMPLNT_NUM"], щоб групувати дані за місяцями та підраховувати кількість злочинів. Метод plot(kind='line') використано, щоб створити лінійний графік, де по горизонталі відкладено місяці, а по вертикалі - кількість злочинів. Цей графік ілюструє зміни кількості злочинів впродовж року.




```
fig= plt.figure(figsize=(15,6))
df.groupby('weekday').count()["CMPLNT_NUM"].plot(kind='line')
plt.xlabel('Days')
plt.xticks([x for x in range(7)])
plt.ylabel('Number of crimes')
plt.ticklabel_format(style='plain', axis='y')
plt.title('Crimes per week days')
```

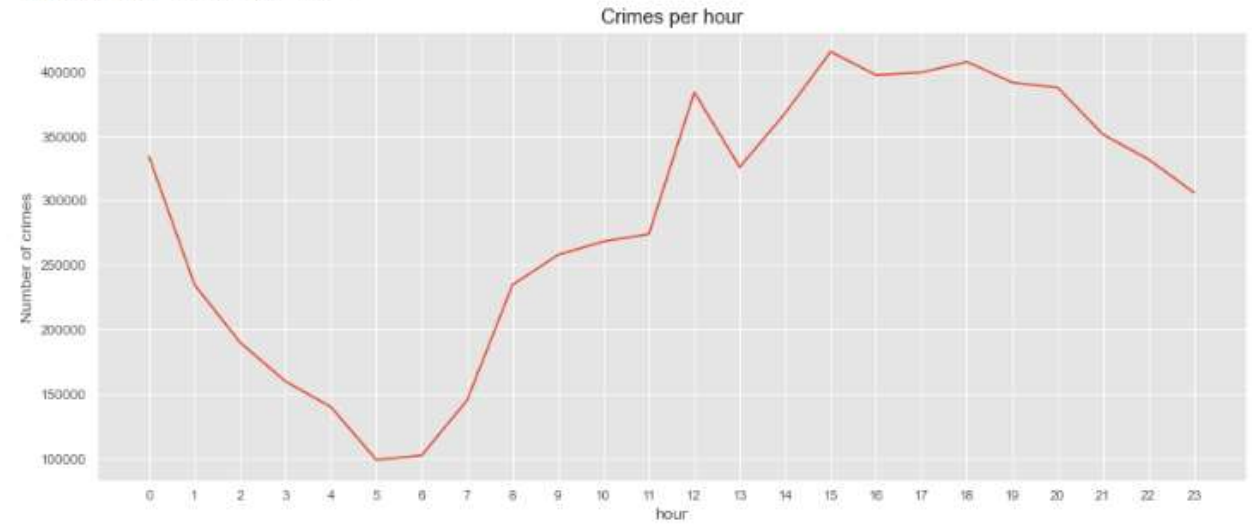
Створюють лінійну діаграму для відображення розподілу злочинів в залежності від днів тижня. Використано groupby('weekday').count()["CMPLNT_NUM"], щоб групувати дані за днями тижня та підраховувати кількість злочинів, plot(kind='line'), щоб створити лінійний графік, де по горизонталі відкладено дні тижня, а по вертикалі - кількість злочинів. Цей графік ілюструє зміни кількості злочинів в залежності від днів тижня:

```
Text(0.5, 1.0, 'Crimes per week days')
```




```
fig= plt.figure(figsize=(15,6))
df.groupby('weekday').count()["CMPLNT_NUM"].plot(kind='line')
plt.xlabel('hour')
plt.xticks([x for x in range(24)])
plt.ylabel('Number of crimes')
plt.title('Crimes per hour')
```

```
Text(0.5, 1.0, 'Crimes per hour')
```



4.2. Розроблення інформаційної системи аналізу рівня злочинності

```
import numpy as np
import pandas as pd
import joblib
import re
import os
from tqdm import tqdm
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
pd.set_option('display.max_rows', 200)
pd.set_option('display.max_columns', 200)
plt.style.use('ggplot')
```

Імпортують необхідні бібліотеки для роботи, такі як numpy, pandas, matplotlib.pyplot, seaborn, joblib. Використовують %matplotlib inline, щоб графіки відображались прямо в блокноті та налаштовують параметри виведення pandas і стиль графіків.

```
from sklearn.ensemble import RandomForestClassifier
from xgboost import XGBClassifier
```

Імпортують класи RandomForestClassifier з бібліотеки Scikit-Learn і XGBClassifier з бібліотеки XGBoost для використання при класифікації. Ці класи є популярними алгоритмами для вирішення задач машинного навчання.

```
from sklearn.model_selection import train_test_split
from sklearn.model_selection import GridSearchCV, RandomizedSearchCV
from sklearn.feature_selection import SelectFromModel
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
from sklearn.metrics import roc_curve, auc
import scikitplot as skplt
```

Імпортують бібліотеки для роботи з класифікацією і оцінки моделей машинного навчання. Це:

- train_test_split - для розбиття даних на тренувальний і тестовий набори;

- GridSearchCV, RandomizedSearchCV - для пошуку оптимальних параметрів моделі;
- SelectFromModel - для вибору важливих ознак;
- classification_report - для виведення звіту про класифікацію, включаючи precision, recall, f1-score;
- confusion_matrix - для створення матриці невідповідностей;
- accuracy_score - для обчислення точності моделі;
- roc_curve, auc - для побудови ROC-кривої та обчислення площі під кривою;
- scikitplot - додаткова бібліотека для візуалізації метрик класифікації.

Ці бібліотеки і методи допоможуть працювати з моделями класифікації та їхніми метриками.

Далі імпортують дані та перетворюють деякі колонки у більш зручний формат. Використовують метод pd.read_csv для імпорту даних з CSV-файлу. print(df.info()) - виводять інформацію про датафрейм, включаючи типи даних та кількість пропущених значень.

df.EVENT_TIME = pd.to_datetime(df.EVENT_TIME).dt.hour - перетворюють стовпець EVENT_TIME в об’єкт datetime, щоб зберегти лише години подій. Це можна буде використати для аналізу за часом доби.

df.LAW_CAT_CD = df['LAW_CAT_CD'].replace(['FELONY','MISDEMEANOR',"VIOLATION"],[2,1,0]) - замінюють значення в стовпці LAW_CAT_CD з текстових на числові (2, 1, 0). В результаті отримали:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6451985 entries, 0 to 6451984
Data columns (total 21 columns):
 #   Column              Dtype
---  ------              -----
 0   EVENT_TIME          object
 1   year                int64
 2   month               int64
 3   day                 int64
 4   Latitude            float64
 5   Longitude           float64
 6   CRM_ATPT_CPTD_CD    object
 7   OFNS_DESC           object
 8   ADDR_PCT_CD         float64
 9   LAW_CAT_CD          object
 10  BORO_NM             object
 11  PREM_TYP_DESC       object
 12  IN_PARK             int64
 13  IN_PUBLIC_HOUSING   int64
 14  IN_STATION          int64
 15  SUSP_AGE_GROUP      object
 16  SUSP_RACE           object
 17  SUSP_SEX            object
 18  VIC_AGE_GROUP       object
 19  VIC_RACE            object
 20  VIC_SEX             object
dtypes: float64(3), int64(6), object(12)
memory usage: 1.0+ GB
None
```

З допомогою методу head виводять перші 5 рядків дата сету:

	EVENT_TIME	year	month	day	Latitude	Longitude	CRM_ATPT_CPTD_CD	OFNS_DESC	ADDR_PCT_CD	LAW_CAT_CD	BORO_NM	PREM_TYP_DESC	IN_PARK	IN_PUBLIC_HOUSING	IN_STATION	SUSP_AGE_GROUP	SUSP_RACE	SU
0	17	2014	9	4	40.685041	-73.921777	COMPLETED	ASSAULT 3 & RELATED OFFENSES	81.0	1	BROOKLYN	STREET	0	0	0	UNKNOWN	UNKNOWN	
1	7	2016	10	12	40.636991	-74.134093	COMPLETED	GRAND LARCENY	121.0	2	STATEN ISLAND	STREET	0	0	0	UNKNOWN	BLACK	
2	13	2012	9	28	40.823876	-73.891863	COMPLETED	GRAND LARCENY	41.0	2	BRONX	STREET	0	0	0	UNKNOWN	WHITE HISPANIC	
3	15	2015	3	24	40.845707	-73.910398	COMPLETED	PETIT LARCENY	46.0	1	BRONX	STREET	0	0	0	UNKNOWN	BLACK	
4	4	2017	5	20	40.763992	-73.828426	COMPLETED	ASSAULT 3 & RELATED OFFENSES	109.0	1	QUEENS	STREET	0	0	0	25-44	WHITE HISPANIC	

```
df.LAW_CAT_CD.value_counts().sort_values(ascending=False)
```

Використано метод value_counts() для підрахунку кількості рядків для кожної унікальної категорії в стовпці LAW_CAT_CD. Результат сортується в порядку спадання. Це використовують для отримання огляду розподілу кількості рядків за різними категоріями правопорушень (FELONY, MISDEMEANOR, VIOLATION):

```
1     3647183
2     1987476
0      817326
Name: LAW_CAT_CD, dtype: int64
```

Використано метод value_counts() для підрахунку кількості рядків для кожної унікальної категорії в стовпці LAW_CAT_CD. Результат сортується

в порядку спадання. Це використовують для отримання огляду розподілу кількості рядків за різними категоріями правопорушень (FELONY, MISDEMEANOR, VIOLATION).

```
zero,one,two = [],[],[]
zero_c,one_c,two_c = 0,0,0
for i in tqdm(df.iterrows()):
    if i[1].LAW_CAT_CD == 0 and zero_c <= 817326:
        zero.append(i[1].values)
        zero_c += 1
    elif i[1].LAW_CAT_CD == 1 and one_c <= 817326:
        one.append(i[1].values)
        one_c += 1
    elif i[1].LAW_CAT_CD == 2 and two_c <= 817326:
        two.append(i[1].values)
        two_c += 1
    if zero_c == 817326 and one_c == 817326 and two_c == 817326:
        break
zero_df = pd.DataFrame(zero, columns=df.columns.values.tolist())
one_df = pd.DataFrame(one, columns=df.columns.values.tolist())
two_df = pd.DataFrame(two, columns=df.columns.values.tolist())
final_df = pd.concat([zero_df,one_df,two_df])
final_df.to_csv("./ny_clean_train_balanced.csv",index=False)
```

Вибирається задана кількість рядків для кожного класу (0, 1, 2), і, як тільки ця кількість досягає 817326, процес завершується.

```
df.info()
df.head()
```

Імпортують збалансований набір даних з CSV-файлу за допомогою pd.read_csv('./ny_clean_train_balanced.csv'). Потім виводять інформацію про датафрейму за допомогою методу df.info() та переглядають перші п'ять рядків за допомогою df.head(). Це допомагає переконатися, що імпорт відбувся успішно та дані мають очікуваний вигляд.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2451980 entries, 0 to 2451979
Data columns (total 21 columns):
 #   Column             Dtype
---  ------             -----
 0   EVENT_TIME         int64
 1   year               int64
 2   month              int64
 3   day                int64
 4   Latitude           float64
 5   Longitude          float64
 6   CRM_ATPT_CPTD_CD   object
 7   OFNS_DESC          object
 8   ADDR_PCT_CD        float64
 9   LAW_CAT_CD         int64
 10  BORO_NM            object
 11  PREM_TYP_DESC      object
 12  IN_PARK            int64
 13  IN_PUBLIC_HOUSING  int64
 14  IN_STATION         int64
 15  SUSP_AGE_GROUP     object
 16  SUSP_RACE          object
 17  SUSP_SEX           object
 18  VIC_AGE_GROUP      object
 19  VIC_RACE           object
 20  VIC_SEX            object
dtypes: float64(3), int64(8), object(10)
memory usage: 392.8+ MB
```

	EVENT_TIME	year	month	day	Latitude	Longitude	CRM_ATPT_CPTD_CD	OFNS_DESC	ADDR_PCT_CD	LAW_CAT_CD	BORO_NM	PREM_TYP_DESC	IN_PARK	IN_PUBLIC_HOUSING	IN_STATION	SUSP_AGE_GROUP	SUSP_RACE
0	17	2011	2	1	40.617674	-73.962811	COMPLETED	HARRASSMENT 2	70.0	0	BROOKLYN	STREET	0	0	0	UNKNOWN	UNKNOWN
1	1	2016	9	27	40.624674	-74.027588	COMPLETED	HARRASSMENT 2	68.0	0	BROOKLYN	RESIDENCE - APT. HOUSE	0	0	0	45-64	WHITE
2	14	2015	12	14	40.836653	-73.907143	COMPLETED	HARRASSMENT 2	42.0	0	BRONX	RESIDENCE - PUBLIC HOUSING	0	1	0	UNKNOWN	BLACK HISPANIC
3	14	2014	8	13	40.594460	-74.065243	COMPLETED	HARRASSMENT 2	122.0	0	STATEN ISLAND	RESIDENCE-HOUSE	0	0	0	65+	WHITE
4	17	2016	10	26	40.774455	-73.932565	COMPLETED	HARRASSMENT 2	114.0	0	QUEENS	RESIDENCE - PUBLIC HOUSING	0	0	0	18-24	BLACK

```
def reduce_mem_usage(df, verbose=True):
    numerics = ['int16', 'int32', 'int64', 'float16', 'float32', 'float64']
    start_mem = df.memory_usage().sum() / 1024**2
    for col in df.columns:
        col_type = df[col].dtypes
        if col_type in numerics:
            c_min = df[col].min()
            c_max = df[col].max()
            if str(col_type)[:3] == 'int':
                if c_min > np.iinfo(np.int8).min and c_max <
np.iinfo(np.int8).max:
                    df[col] = df[col].astype(np.int8)
                elif c_min > np.iinfo(np.int16).min and c_max <
np.iinfo(np.int16).max:
                    df[col] = df[col].astype(np.int16)
                elif c_min > np.iinfo(np.int32).min and c_max <
np.iinfo(np.int32).max:
                    df[col] = df[col].astype(np.int32)
```

```
                elif c_min > np.iinfo(np.int64).min and c_max <
np.iinfo(np.int64).max:
                    df[col] = df[col].astype(np.int64)
            else:
                if c_min > np.finfo(np.float16).min and c_max <
np.finfo(np.float16).max:
                    df[col] = df[col].astype(np.float16)
                elif c_min > np.finfo(np.float32).min and c_max <
np.finfo(np.float32).max:
                    df[col] = df[col].astype(np.float32)
                else:
                    df[col] = df[col].astype(np.float64)
    end_mem = df.memory_usage().sum() / 1024**2
    print('Memory usage after optimization is: {:.2f} MB'.format(end_mem))
    print('Decreased by {:.1f}%'.format(100 * (start_mem - end_mem) /
start_mem))
    return df
df = reduce_mem_usage(df)
```

Функція reduce_mem_usage використовується для оптимізації використання пам’яті. Вона перебирає числові стовпці у датафреймі та перетворює їх типи.

```
feature_lst=['EVENT_TIME','ADDR_PCT_CD', 'month', 'day', 'Latitude',
       'Longitude', 'BORO_NM',"WEEKDAY",
       'IN_PARK', 'IN_PUBLIC_HOUSING', 'IN_STATION', 'VIC_AGE_GROUP',
       'VIC_RACE', 'VIC_SEX','LAW_CAT_CD']

df_sel=df[feature_lst].copy()
df_sel.info()
df_sel.head()
```

Вибирають лише ті стовпці, які залишаться значущими для моделювання. Вибрані ознаки включають інформацію про час події, місце, різні категорії та характеристики жертви, а також цільову змінну LAW_CAT_CD.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2451980 entries, 0 to 2451979
Data columns (total 14 columns):
 #   Column             Dtype
---  ------             -----
 0   EVENT_TIME         int8
 1   ADDR_PCT_CD        float16
 2   month              int8
 3   day                int8
 4   Latitude           float16
 5   Longitude          float16
 6   BORO_NM            object
 7   IN_PARK            int8
 8   IN_PUBLIC_HOUSING  int8
 9   IN_STATION         int8
 10  VIC_AGE_GROUP      object
 11  VIC_RACE           object
 12  VIC_SEX            object
 13  LAW_CAT_CD         int8
dtypes: float16(3), int8(7), object(4)
memory usage: 105.2+ MB
```

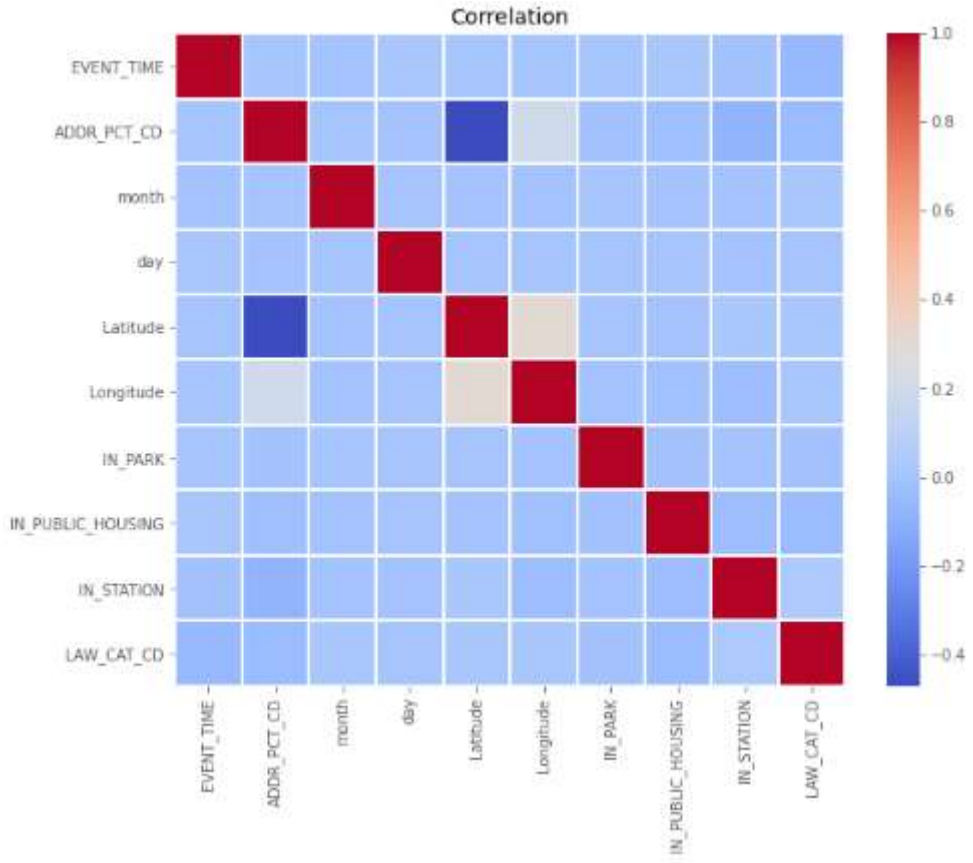
	EVENT_TIME	ADDR_PCT_CD	month	day	Latitude	Longitude	BORO_NM	IN_PARK	IN_PUBLIC_HOUSING	IN_STATION	VIC_AGE_GROUP	VIC_RACE	VIC_SEX	LAW_CAT_CD
0	17	70.0	2	1	40.62500	-73.9375	BROOKLYN	0	0	0	18-24	WHITE	F	0
1	1	68.0	9	27	40.62500	-74.0000	BROOKLYN	0	0	0	65+	WHITE	M	0
2	14	42.0	12	14	40.84375	-73.9375	BRONX	0	1	0	25-44	WHITE HISPANIC	F	0
3	14	122.0	8	13	40.59375	-74.0625	STATEN ISLAND	0	0	0	25-44	WHITE	M	0
4	17	114.0	10	26	40.78125	-73.9375	QUEENS	0	0	0	45-64	BLACK	F	0

```
print(df_sel.shape)
df_sel.LAW_CAT_CD.value_counts().sort_values(ascending=False)
(2451980, 14)
1    817327
2    817327
0    817326
Name: LAW_CAT_CD, dtype: int64
```

Використовують print(df_sel.shape) для виведення розміру нового датафрейму і метод df_sel.LAW_CAT_CD.value_counts().sort_values(ascending=False) для подальшої перевірки балансу категорій в цільовій змінній.

```
corr = df_sel.corr()
plt.figure(figsize = (10,8))
sns.heatmap(corr, cmap = "coolwarm", linewidth = 2, linecolor = "white")
plt.title("Correlation")
plt.show()
```

Далі використовують бібліотеку Seaborn для побудови теплової карти кореляції між змінними у датафреймі. Такі візуалізації можуть бути корисними для виявлення кореляцій між ознаками та визначення, які з них можуть бути важливими для моделі. Це спосіб візуалізувати взаємозв’язки між різними ознаками у наборі даних.




```
df_state_dummy = pd.get_dummies(df_sel)
df_state_dummy.info()
```

Використовують метод pd.get_dummies(df_sel) для створення змінних для категоріальних ознак у дата фреймі df_sel. Цей метод розширює категоріальні змінні у бінарні (0 або 1) за допомогою техніки кодування one-hot encoding. В результаті отримують новий дата фрейм df_state_dummy, де кожна категоріальна змінна розширена в бінарні змінні.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2451980 entries, 0 to 2451979
Data columns (total 35 columns):
 #   Column                                  Dtype
---  ------                                  -----
 0   EVENT_TIME                              int8
 1   ADDR_PCT_CD                             float16
 2   month                                   int8
 3   day                                     int8
 4   Latitude                                float16
 5   Longitude                               float16
 6   IN_PARK                                 int8
 7   IN_PUBLIC_HOUSING                       int8
 8   IN_STATION                              int8
```

```
 9   LAW_CAT_CD                                int8
 10  BORO_NM_BRONX                             uint8
 11  BORO_NM_BROOKLYN                          uint8
 12  BORO_NM_MANHATTAN                         uint8
 13  BORO_NM_QUEENS                            uint8
 14  BORO_NM_STATEN ISLAND                     uint8
 15  BORO_NM_UNKNOWN                           uint8
 16  VIC_AGE_GROUP_18-24                       uint8
 17  VIC_AGE_GROUP_25-44                       uint8
 18  VIC_AGE_GROUP_45-64                       uint8
 19  VIC_AGE_GROUP_65+                         uint8
 20  VIC_AGE_GROUP_<18                         uint8
 21  VIC_AGE_GROUP_UNKNOWN                     uint8
 22  VIC_RACE_AMERICAN INDIAN/ALASKAN NATIVE   uint8
 23  VIC_RACE_ASIAN / PACIFIC ISLANDER         uint8
 24  VIC_RACE_BLACK                            uint8
 25  VIC_RACE_BLACK HISPANIC                   uint8
 26  VIC_RACE_OTHER                            uint8
 27  VIC_RACE_UNKNOWN                          uint8
 28  VIC_RACE_WHITE                            uint8
 29  VIC_RACE_WHITE HISPANIC                   uint8
30  VIC_SEX_D                                 uint8
 31  VIC_SEX_E                                 uint8
 32  VIC_SEX_F                                 uint8
 33  VIC_SEX_M                                 uint8
 34  VIC_SEX_U                                 uint8
dtypes: float16(3), int8(7), uint8(25)
memory usage: 88.9 MB
```

Виводять перші п’ять рядків нового датафрейму:

	EVENT_TIME	ADDR_PCT_CD	month	day	Latitude	Longitude	IN_PARK	IN_PUBLIC_HOUSING	IN_STATION	LAW_CAT_CD	BORO_NM_BRONX	BORO_NM_BROOKLYN	BORO_NM_MANHATTAN	BORO_NM_QUEENS	BORO_NM_STATEN ISLAND	BORO_NM_UNKNOWN
0	17	70.0	2	1	40.62500	-73.9375	0	0	0	0	0	1	0	0	0	0
1	1	68.0	9	27	40.62500	-74.0000	0	0	0	0	0	1	0	0	0	0
2	14	42.0	12	14	40.84375	-73.9375	0	1	0	0	1	0	0	0	0	0
3	14	122.0	8	13	40.59375	-74.0625	0	0	0	0	0	0	0	0	1	0
4	17	114.0	10	26	40.78125	-73.9375	0	0	0	0	0	0	0	1	0	0

```
df=df_state_dummy
target='LAW_CAT_CD'
y = df[target]
```

Далі використовують метод df = df_state_dummy для призначення згенерованого датафрейму назад до оригінальної змінної df. Визначають цільову змінну для прогнозу як target=LAW_CAT_CD. Рядок y = df[target]

вказує, що y буде містити цільову змінну для прогнозу, тобто категорії правопорушень (LAW_CAT_CD).

```
y.unique()
y.value_counts()
1    817327
2    817327
0    817326
Name: LAW_CAT_CD, dtype: int64

X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2,shuffle=True, random_state=21, stratify=y)
```

Використовують train_test_split для розбиття набору даних на тренувальний та тестовий набори.

- X - масив ознак, який містить усі стовпці, крім цільової змінної.

- y - масив цільової змінної, тобто категорії правопорушень.

- test_size=0.2 - вказує, що 20 % даних буде використано для тестування.

- shuffle=True - вказує, що дані будуть перемішані перед розбиттям.

```
def plot_cm(y_pred,y_test,algorithm,figure_name):
    mat_RF = confusion_matrix(y_pred,y_test)
    plt.figure(figsize=(16,4))
    sns.heatmap(mat_RF, square=True, annot=True, fmt='d',
cbar=False,xticklabels=[0,1,2],yticklabels=[0,1,2])
    plt.xlabel('True labels')
    plt.ylabel('predicted labels')
    plt.title(algorithm)
    plt.savefig(figure_name)
```

Функція plot_cm використовує бібліотеку Seaborn для побудови теплової карти матриці невідповідностей. Вона приймає прогнозовані y_pred і фактичні y_test значення, назву алгоритму та ім’я файлу для збереження фігури.

```
def plot_roc(y_test, model, figure_name):
    pl = skplt.metrics.plot_roc(y_test, model.predict_proba(X_test),
figsize=(12,6))
    plt.show()
    pl.figure.savefig(figure_name)
```

Функція plot_roc використовує бібліотеку scikit-plot для побудови кривої ROC (Receiver Operating Characteristic). Вона приймає фактичні значення y_test, модель та ім’я файлу для збереження фігури.

```
def save_model(model, model_name,is_tree=False):
    joblib.dump(model.estimators_[0] if is_tree else
model,f'{model_name}.joblib')
    print(f"Model size: {np.round(os.path.getsize(f'{model_name}.joblib') /
1024 / 1024, 2) } MB")
clf=RandomForestClassifier(n_estimators=100,n_jobs=-1,verbose=1)
clf.fit(X_train,y_train)
y_pred=clf.predict(X_test)
acc_rf = accuracy_score(y_test, y_pred)
class_report = classification_report(y_test, y_pred)
print(class_report)
```

Використано Random Forest Classifier з бібліотеки Scikit-Learn для навчання моделі та оцінки її ефективності. Основні кроки включають наступне:

1. Створення класифікатора:

Вказують 100 дерев n_estimators=100, використовують всі доступні ядра для паралельної роботи (`n_jobs=-1`) та включають вивід під час навчання verbose=1.

2. Навчання моделі: навчають модель за допомогою тренувальних даних.

3. Прогнозування та оцінка моделі: використовують навчену модель для прогнозування на тестових даних, а потім отримують метрики для оцінки ефективності моделі.

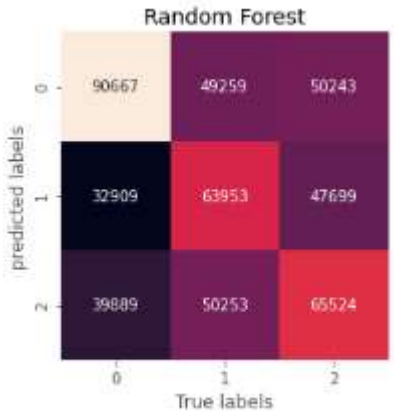
4. Отримання метрик точності та виведення звіту: отримують точність та звіт про класифікацію, який містить інформацію про різні метрики якості. Результат:

```
precision    recall  f1-score   support

           0       0.48      0.55      0.51    163465
           1       0.44      0.39      0.42    163465
           2       0.42      0.40      0.41    163466

    accuracy                           0.45    490396
   macro avg       0.45      0.45      0.45    490396
```

```
weighted avg       0.45      0.45      0.45    490396
```

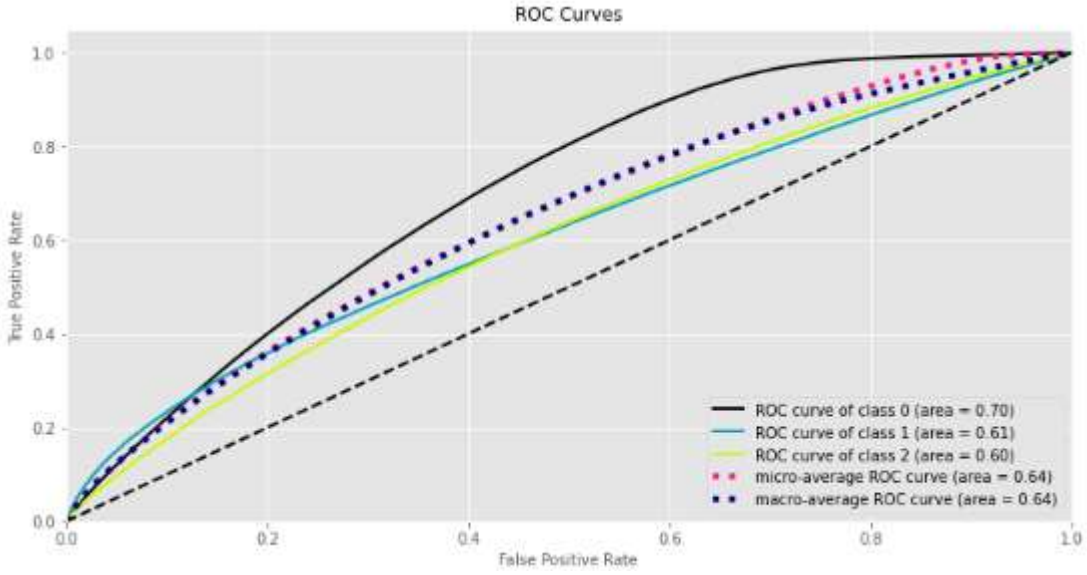
```
plot_cm(y_pred,y_test,"Random Forest","cm_random_forest.pdf")
```

Викликають функцію plot_cm для побудови матриці невідповідностей для моделі Random Forest. Це буде корисно для візуалізації того, наскільки модель добре класифікує дані.




```
plot_roc(y_test,clf,"roc_random_forest.pdf")
```

Викликають функцію plot_roc для побудови кривої ROC для моделі Random Forest. Крива ROC вказує на ефективність моделі в розрізі чутливості та специфічності.




```
save_model(clf,"random_forest",True)
```

Викликають функцію save_model. Згідно з аргументами вона може зберегти модель Random Forest під ім’ям random_forest.pkl.

```
from the columns name
regex = re.compile(r"\[|\]|<", re.IGNORECASE)
X_train.columns = [regex.sub("_", col) if any(x in str(col) for x in
set(('[', ']', '<'))) else col for col in X_train.columns.values]

params = {
            'objective':'multi:softmax',
```

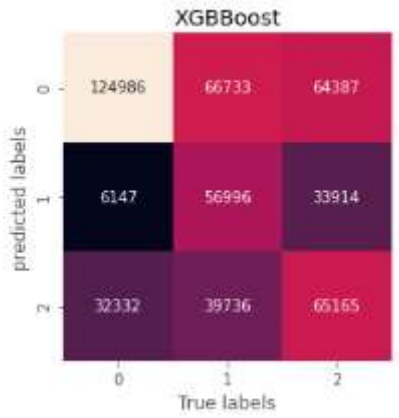
```
            'max_depth': 10,
            'alpha': 10,
            'learning_rate': 0.1,
            'n_estimators':100,
            'use_label_encoder':False
        }

xgb_clf = XGBClassifier(**params)

xgb_clf.fit(X_train, y_train)

y_pred = xgb_clf.predict(X_test)
accuracy = accuracy_score(y_pred, y_test)
print('XGBBoost Model accuracy score:
{0:0.4f}'.format(accuracy_score(y_test, y_pred)))

class_report = classification_report(y_test, y_pred)
print(class_report)
```

Використовують XGBClassifier з бібліотеки XGBoost та навчають модель з використанням гіперпараметрів. Потім оцінюють ефективність моделі та отримують звіт про класифікацію.

```
precision    recall  f1-score   support

           0       0.49      0.76      0.60    163465
           1       0.59      0.35      0.44    163465
           2       0.47      0.40      0.43    163466

    accuracy                           0.50    490396
   macro avg       0.52      0.50      0.49    490396
weighted avg       0.52      0.50      0.49    490396
```

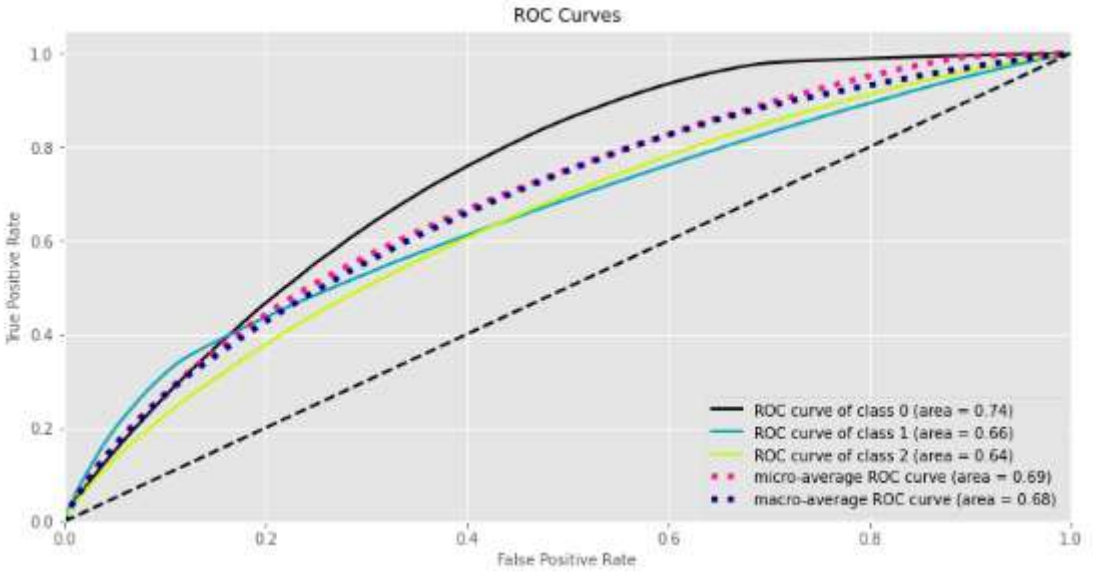
```
plot_cm(y_pred, y_test, "XGBBoost", 'cm_XGBBoost.pdf')
```

Викликають функцію plot_cm для побудови матриці невідповідностей для моделі XGBoost. Це буде корисно для візуалізації того, наскільки модель добре класифікує дані та для порівняння з результатами моделі Random Forest.




```
plot_roc(y_test, lbm_clf, "roc_XGBBoost.pdf")
```

Викликають функцію plot_roc для побудови кривої ROC для моделі XGBoost. Ця крива ROC вказує на ефективність моделі XGBoost по чутливості та специфічності.



ВИСНОВКИ ДО РОЗДІЛУ 4

В даному розділі приведено дані про принципи побудови алгоритму математичної моделі та підходів до…. На основі приведеної математичної моделі спроектовано інформаційну систему. Проведені дослідження та моделювання підтверджують достовірність розробленої математичної моделі інформаційної системи та можливості по ….

РОЗДІЛ 5. РОЗРОБЛЕННЯ СТАРТАП ПРОЕКТУ

5.1. Опис проекту інформаційної системи

Розгянемо процес створення інформаційної карти. На ній потрібно представити параметри проекту та скільки грошей потрібно на його виконання.

Табл. 5.1. Структура проекту інформаційної системи

Назва номінації	Інформаційна система, розроблена на мові програмування Python
Назва проекту	Інтелектуальна система аналізу криміногенної ситуації
Назва ВНЗ, факультету, спеціальності	НЛТУ, кафедра комп'ютерних наук, 122 «Комп'ютерні науки»
Прізвище, ім'я, по-батькові	Лакуста Андрій Васильович
Цілі і задачі проекту	Мета проекту – розробка та реалізація інформаційно-аналітичної системи для можливості прогнозування рівня злочинності. Задачі проекту: • проаналізувати існуючі системи прогнозування рівня злочинності; • розробити математичну модель інформаційної системи рівня злочинності; • реалізувати програмну модель інформаційної системи рівня злочинності мовою Python; • провести дослідження параметрів даної інформаційної системи,

	представити її результати роботи.
Короткий зміст проекту	В роботі приводиться опис найбільш відомих моделей прогнозування, досліджується взаємозв'язок кількості правопорушень та різних зовнішніх факторів (дня тижня або погодних умов). На підставі отриманих даних приведено графіки, які відображають найбільш небезпечний час доби, дні тижня та місяці в році, в які необхідно посилити поліцейський контроль на вулицях міста. Окрім цього, розроблено інтерактивну карту, яка дозволить знайти найбільш небезпечні райони та вулиці міста. На основі отриманих закономірностей розроблено модель машинного навчання для прогнозування рівня правопорушень, які враховують як історичні дані, так і різні зовнішні фактори. Приводяться оцінки точності отриманих результатів, за якими можна зробити висновок про якість прогнозів. В цій роботі отримано прогнози криміногенного рівня злочинності в місті Нью-йорк, на основі яких можуть бути визначені несприятливі дні, в яких число правопорушень значно перевищує середній рівень.

5.2. Інвестиційна привабливість стартапу

Оцінка інвестиційної привабливості продукту саме по собі нетривіальне завдання, тим більше коли справа стосується стартапів. На етапі планування нового проекту важливо знати не тільки можливий вплив зовнішніх факторів галузі, але й яких трудовитрат вимагає проект, за який термін його буде реалізовано і, головне, скільки коштів на це потрібно. Щоб уникнути провалів, проводиться попередня оцінка стартапу та навколишнього середовища, на основі отриманої оцінки приймається рішення про реалізацію проекту або про його відхилення. Існує багато різних методик складання точного плану бюджету. Усі вони мають свої плюси та мінуси. Щоб вибрати конкретний спосіб оцінки інвестиційної привабливості стартапів, необхідно розібратися в існуючих методах їх оцінки.

Вкладення коштів у стартапи проектів є вигідним, тобто стартапи мають інвестиційну привабливість з економічної точки зору. Це пов’язано зі швидким розвитком бізнесу в інтернеті та посиленим інтересом до цієї галузі економіки, а також зі збільшенням споживачів у мережі інтернет. Оцінка фінансово-інвестиційної привабливості проводиться за допомогою математичних інструментів. Однак формування методології оцінки знаходиться в стані, що зароджується. Одним із головних математичних інструментів при оцінці інвестиційної привабливості є коефіцієнт норми прибутковості, який показує, у скільки разів у майбутньому збільшаться кошти, проінвестовані сьогодні. Активом виступають компанії, які лише розпочинають свою діяльність. Для оцінки розмірів доходів інвестору від даних компаній у майбутньому, необхідно оцінити, якого фінансового результату досягне стартап, тобто скільки коштуватиме сама компанія через певний часовий період (розрахувати прибутковість інвестицій).

Для оцінки вартості компаній існують три підходи: прибутковий, витратний та порівняльний. Прибутковий виходить з розрахунках доходів підприємства у майбутньому. Розрахунок майбутніх доходів провадиться

на основі аналізу минулих грошових потоків, активів компанії, її фінансових показників. Для порівняльного підходу необхідна наявність над ринком порівняних фірм, котрі займаються ідентичною діяльністю для можливості зібрати необхідну інформацію з ринку. Витратний підхід виходить з обліку вартості підприємства як ринкової вартості всіх її активів з відрахуванням довгострокових зобов'язань.

Проте, вищевикладені три основні підходи слабо застосовні щодо оцінки вартості стартапів.

– Стартап не має історії: оскільки компанії тільки починають здійснювати свої перші економічні транзакції, а деякі проекти знаходяться лише на стадії ідеї, неможливо будувати будь-які фінансові прогнози та стратегії через відсутність необхідного аналізу господарської діяльності, як і не надається можливим оцінка ринкової вартості поточних активів через їхню відсутність. Виходячи з цього, дохідний та витратний підходи виявляються важкозастосовними;

- Стартап відрізняє новизна: найчастіше метою створення стартапів служить заповнення будь-якої порожньої ніші або створення нової. Крім того, стартапи проектів використовують високі технології та новітні методики, а це, у свою чергу, означає, що знайти аналогічні компанії практично неможливо, тобто порівняльний метод є неефективним.

Таким чином, враховуючи наведені вище особливості, завдання оцінки вартості стартапів ускладнюється. Очевидно, що до стартапів не можна застосовувати класичні методи оцінки, потрібна їхня модифікація.

Зважаючи на відсутність універсального методу оцінки інвестиційної привабливості стартапу та розрахунку точної вартості проекту, кожна існуюча оцінка стартапу містить у собі певний відсоток суб'єктивності, а найчастіше застосовні методи – експертні, які ґрунтуються на досвіді інвестора чи аналітика. Однак існує кілька підходів, які дозволяють розрахувати так звану базову цифру. Розрахунки ґрунтуються більшою

мірою на прогнозних значеннях показників, що входять до них, оскільки реальних грошових потоків стартап не виробляє.

5.3. Джерела фінансування стартапів

Кожен стартап потребує доступу до капіталу, чи буде цей капітал спрямований на фінансування досліджень і розробок, придбання інвентарю або виплати зарплати. Більшість підприємців, які стикаються з проблемою нестачі вільних коштів, починають думати про банківські кредити як основне джерело фінансування, проте існують й інші оригінальні способи отримання та максимізації небанківського фінансування. Серед таких варіантів фінансування виділяються: краудсорсинг, різні компанії та приватні особи, зацікавлені у вкладенні коштів у стартапи. Підприємства можуть уникнути боргів із високими відсотками за банківськими кредитами на забезпечення фінансування за рахунок інвесторів.

Оскільки кількість стартапів стає дедалі більше, новостворена компанія має бути винахідливою з метою залучення інвестицій. Створення бренду та демонстрація досяжних цілей – це лише початок. Однак є багато різних тонкощів, які слід врахувати при залученні потенційних інвесторів та отриманні фінансування.

1. Підготувати фінансову модель, яка розповідатиме свою власну історію.

Найчастіше інвестори самі шукають стартапи, які потребують фінансування, вони перевіряють, які компанії зможуть правильно використати будь-який капітал, який вони одержують. Стартап - це новостворена компанія, яка не має солідної історії за плечима, що дозволяє потенційним інвесторам судити про компанію. В даний час багато нових компаній володіють величезною енергією, ідеями, проте голі ідеї не привабливі для потенційних інвесторів. Інвестори зацікавлені у примноженні свого капіталу. У ситуації, що склалася, якісна фінансова модель розповідатиме свою власну історію створенню довірчого стартапу.

Поряд із підготовкою бізнес-плану важливо продумати стратегію виходу на ринок, вибір ніші, сегмент клієнтів, яких компанія хоче залучити.

Фінансова модель може містити високу, але досяжну мету, показувати потенційним інвесторам основні операційні витрати, як будуть використані кошти: на розширення виробництва або збільшення персоналу, дослідження та розробку або відкриття нових офісів. Якщо говоримо про стартап, який уже проіснував якийсь час, перш ніж знайти потенційного інвестора, то в даному випадку слід показати, як компанія використовувала свій бюджет, перш ніж знайшла потенційних інвесторів.

2. Обґрунтувати результати, які показані у фінансовій моделі. У світі важко когось здивувати, так як немає великої кількості абсолютно інноваційних продуктів. Більшість стартапів розробляють продукти та рішення, які є надзвичайно актуальними для поточних суспільних потреб. У таких умовах корисно показати інвесторам, яким затребуваний стартап буде у майбутньому, як він боротиметься за клієнтів.

В умовах економічних умов, що часто змінюються, стартап повинен бути гнучким для того, щоб залишитися на плаву. Коли немає конкретного та чіткого плану дій, стартап може спершу згенерувати успіх, але незабаром потрапити в період краху. На успіх стартапу, крім продуманої продуктової стратегії розвитку, дуже важливий вплив має маркетингова стратегія. Кошти, витрачені на маркетинг, не повернуться відразу ж, але зрозуміло, що маркетинг необхідний. Маркетингові стратегії повинні включати програми в настільки популярних в даний час соціальних мережах і спільнотах, щоб по-справжньому зрозуміти, як компанії можуть підлаштуватися під своїх клієнтів, до їхніх майбутніх потреб.

3. Стартапи мають шукати інвесторів, а не чекати, коли хтось зверне увагу на компанію випадково. Інвестори не мають достатньої кількості часу, щоб досканально вивчити бізнес-план компанії та виявити ключові проблеми. Презентація стартапу для інвестора має бути короткою, яскравою та водночас інформативною для того, щоб отримати фінансову

підтримку. У презентації компанія може чітко визначити, чому вона потребує фінансування і як вона має намір використовувати кошти та виділяти на свої операційні потреби.

4. Продемонструвати успішність. Стартап має сам заявити про себе. Щоб бути на вустах компанії, слід брати участь у різноманітних маркетингових заходах, конференціях. Історії успіху здатні виділити стартап серед інших компаній займаної ніші, особливо це стосується галузей, які буквально затопили компанії новачки.

Невдала спроба збільшує шанси на успіх у майбутньому. Підприємці, які успішно розмістили акції та отримали фінансування при першому раунді, мають 30 % ймовірності успіху при вторинному розміщенні. Компанії, які вперше розміщують акції, можуть розраховувати на 18 % успіху, а компанії, які вже зазнали невдачі, можуть бути успішними цього разу з 20 % ймовірності. Виходить, що новостворена компанія (стартап) має менше шансів на успіх у порівнянні з тією, що вже намагалася розмістити свої акції, але зазнала невдачі. Ключовим моментом є важливий досвід, отриманий компанією при спробі публічного розміщення акцій.

Підприємці, які постійно працюють над новими ідеями, більш привабливі для інвесторів. Підприємці, які готові паралельно працювати над кількома інноваційними продуктами, мають вищі показники успішності. Згідно зі статистикою, 45 % нових підприємств отримають перший раунд фінансування на ранній стадії і майже 60 % підприємців отримають перший раунд фінансування на ранній стадії, коли це їхнє друге розміщення акцій.

Оцінка вартості стартапу значною мірою визначається і на основі якісних ознак. Оцінити компанію, яка не приносить дохід, складно. Є багато рішень для побудови процесу оцінки, але навіть після того, як взяли всі деталі до уваги, остаточна оцінка компанії буде схожа одночасно на мистецтво і науку. Ухвалення рішення про вартість стартапу схоже на оцінку витвору мистецтва: є основи та принципи, але в результаті процес

оцінки виглядатиме як обґрунтоване припущення. Але, не вклавши гроші в компанію, на жаль, не вдасться легко дізнатися, чи була оцінка компанії вірною або помилковою.

ВИСНОВКИ ДО РОЗДІЛУ 5

Представлено структуру інформаційної системи. В ній приведено назву проекту, мету проекту та його задачі. В роботі приводиться опис найбільш відомих моделей прогнозування, досліджується взаємозв’язок кількості правопорушень та різних зовнішніх факторів (дня тижня або погодних умов).

Приведено дані про інвестиційну привабливість стартапу. На етапі планування нового проекту важливо знати не тільки можливий вплив зовнішніх факторів галузі, але й яких трудовитрат вимагає проект, за який термін його буде реалізовано і, головне, скільки коштів на це потрібно. Показано, що вкладення коштів у стартапи проектів є вигідним, тобто стартапи мають інвестиційну привабливість з економічної точки зору. Для оцінки вартості компаній преставлено три підходи. Представлено дані про джерела фінансування стартапів. Серед таких варіантів фінансування виділяють краудсорсинг, різні компанії та приватні особи, які будуть зацікавлені у вкладення коштів у стартапи.

ВИСНОВКИ

У дипломній роботі наведено докладний аналіз рівня злочинності в США, м. Нью-йорк. Був проаналізований зв’язок рівня злочинності та різних зовнішніх факторів, таких як погодні умови, день тижня, номер дня в році. Приведено візуалізацію даних, яка дозволяє побачити цікаві закономірності. На основі географічних даних про злочинність отримано інтерактивні карти, на яких можна знайти найбільш небезпечні вулиці та райони міста.

На підставі виявлених закономірностей розроблено модель для прогнозування рівня злочинності для таких злочинів, як напад, побої, кражі зі взломом, спричинення збитку, мошенництва, грабежу та воровства. Модель розроблена на основі статистичних даних за 2001-2021 роки. Для її проектування використовувалися алгоритми машинного навчання, які враховують як історичні дані про злочини, так й інші зовнішні фактори. Приведено порівняльний аналіз алгоритмів. Ця модель показала достатньо високу точність. Але найкращий результат дав алгоритм регресії на основі випадкових лісів.

Результати, які отримані в рамках цього дослідження, можуть виявитися корисними для прогнозування рівня злочинності в інших містах. Дані методи прогнозування часових рядів можуть бути корисні для зниження рівня злочинності, так як дозволяють передбачати несприятливі періоди, в яких необхідно посилювати патрулювання вулиць міста.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Басюк Т.М., Литвин В.В., Захарія Л.М., Кунанець Н.Е.. Машинне навчання: навчальний посібник / – Львів: Видавництво «Новий Світ - 2000», 2019. – 335 с.

2. Баган Т.Г.. Методи машинного навчання при проектуванні автоматизованих систем керування: навчальний посібник / – КПІ ім. Ігоря Сікорського. – Електронні текстові дані. – Київ: КПІ імені Ігоря Сікорського, 2021. – 28 с.

3. Олещенко Л.М. Машинне навчання: комп’ютерний практикум з дисципліни «Машинне навчання» / – КПІ ім. Ігоря Сікорського. – Електронні текстові дані. – Київ: КПІ ім. Ігоря Сікорського, 2022. – 92 с.

4. Кононова К. Ю. Машинне навчання: методи та моделі: / – Харків: ХНУ імені В. Н. Каразіна, 2020. – 301 с.

5. Лубко Д.В., Шаров С.В. Методи та системи штучного інтелекту: навчальний посібник / – Мелітополь: ФОП Однорог Т.В., 2019. – 264 с.

6. Копей В.Б. Мова програмування Python для інженерів і науковців. Навчальний посібник / – Івано-Франківськ : ІФНТУНГ, 2019. – 272 с.

7. Ситник В.Ф., Краснюк М.Т. Інтелектуальний аналіз даних (дейтамайнінг) / – Київ: КНЕУ, 2007. – 376 с.

8. Щербаковський М.Г., Пашнєв Д.В. Розслідування комп’ютерних злочинів / – Х.: ХНУВС, 2010. – 112 с.

9. Фролова О.Г. Злочинність і система кримінальних покарань / – К.: "АртЕк", 1997. – 208 с.

10. Литвак О.М. Держава і злочинність. Монографія / – К.: Атіка, 2004 . – 304 с.

11. Іванов Ю.Ф., Джужа О.М. Кримінологія. Навчальний посібник / – К.: Паливода А. В., 2006. – 264 с.

12. Zollanvari Amin. Machine Learning with Python: Theory and Implementation / – Springer, 2023. – 457 p.

13. Zhou Hong. Learn Data Mining Through Excel: A Step-by-Step Approach for Understanding Machine Learning Methods / – Apress, 2020. – 223 p.

14. Liu G.R. Machine Learning with Python: Theory and Applications / – Singapore: World Scientific Publishing Company, 2022. – 692 p.

ДОДАТКИ

ДОДАТОК А

1.ipynb

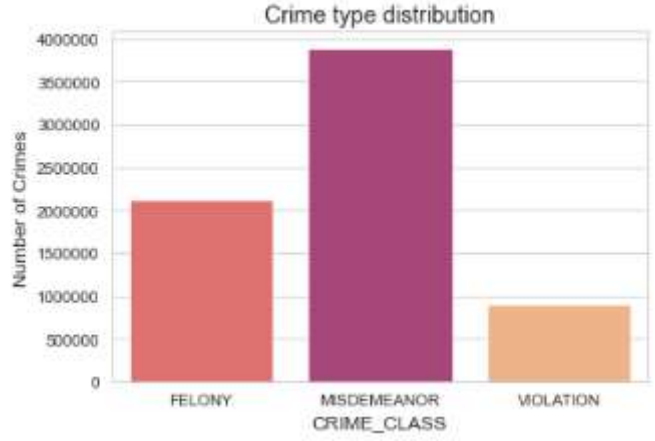
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
plt.style.use('ggplot')
import seaborn as sns
import folium
import folium.plugins as plugins
from folium.features import ClickForMarker
from folium.plugins import HeatMap
from branca.element import Template, MacroElement

df = pd.read_csv('ny_clean_all.csv')

df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6882148 entries, 0 to 6882147
Data columns (total 23 columns):
 #   Column             Dtype
---  ------             -----
 0   CMPLNT_NUM         int64
 1   year               int64
 2   month              int64
 3   day                int64
 4   weekday            object
 5   hour               int64
 6   Latitude           float64
 7   Longitude          float64
 8   CRM_ATPT_CPTD_CD   object
 9   OFNS_DESC          object
 10  ADDR_PCT_CD        float64
 11  CRIME_CLASS        object
 12  BORO_NM            object
 13  PREM_TYP_DESC      object
 14  IN_PARK            int64
 15  IN_PUBLIC_HOUSING  int64
```

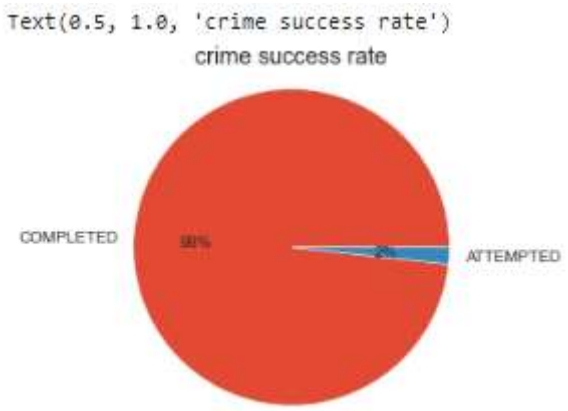
```
 16  IN_STATION         int64
 17  SUSP_AGE_GROUP     object
 18  SUSP_RACE          object
 19  SUSP_SEX           object
 20  VIC_AGE_GROUP      object
 21  VIC_RACE           object
 22  VIC_SEX            object
dtypes: float64(3), int64(8), object(12)
memory usage: 1.2+ GB
```

```
df2=df.groupby(['CRIME_CLASS'])['CMPLNT_NUM'].count().reset_index()
sns.set_style('whitegrid')
palette = sns.color_palette("magma",5)
data = df.groupby(['CRIME_CLASS'])['CMPLNT_NUM'].size()
rank = data.argsort().argsort()
g=sns.barplot(x='CRIME_CLASS',y='CMPLNT_NUM',data=df2,palette=np.array(pale
tte[::-1])[rank])
plt.ylabel('Number of Crimes')
plt.ticklabel_format(style='plain', axis='y')
plt.title("Crime type distribution")
```

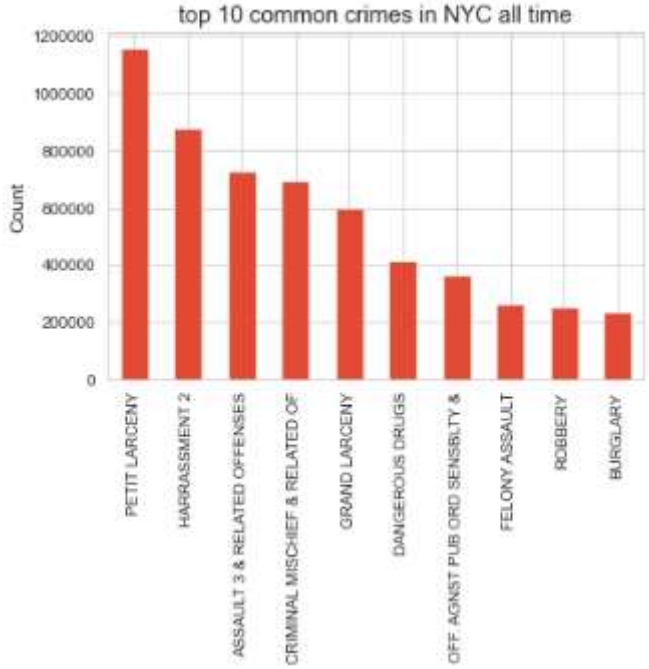
```
Text(0.5, 1.0, 'Crime type distribution')
```



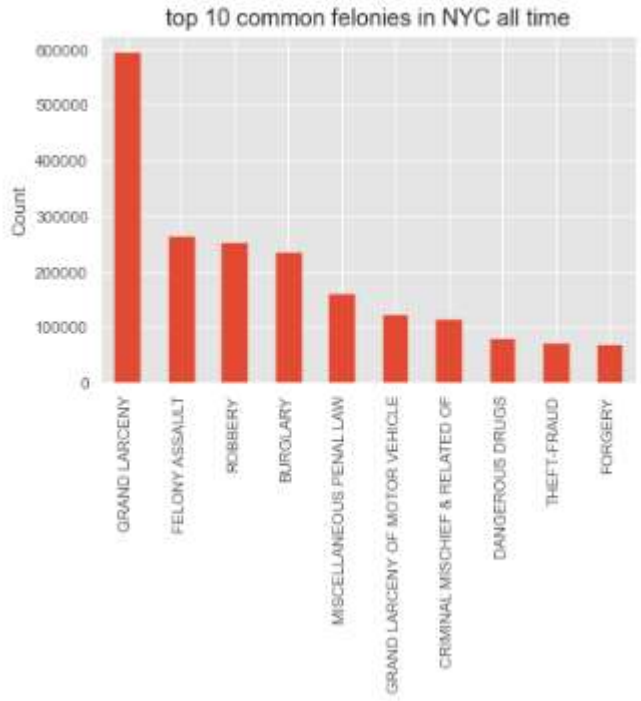

```
plt.pie(df['CRM_ATPT_CPTD_CD'].value_counts(normalize=True).round(4),labels
=df['CRM_ATPT_CPTD_CD'].unique(),autopct='%.0f%%')
plt.axis('equal')
plt.title('crime success rate')
```




```
df['OFNS_DESC'].value_counts()[:10].plot.bar()
plt.ylabel('Count')
plt.title('top 10 common crimes in NYC all time')
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```

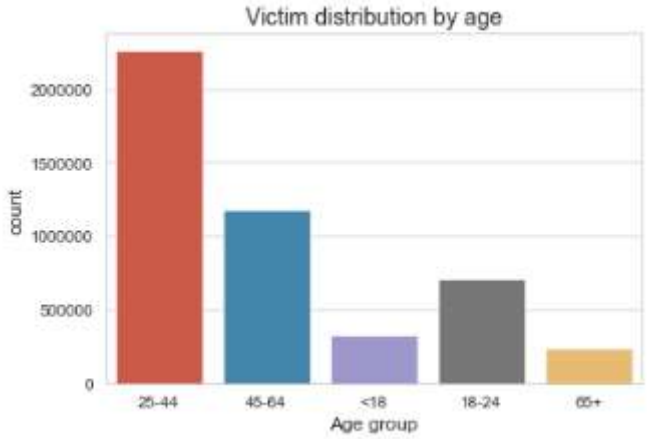



```
df[df['CRIME_CLASS']=='FELONY']['OFNS_DESC'].value_counts()[:10].plot.bar()
plt.ylabel('Count')
plt.title('top 10 common felonies in NYC all time')
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```

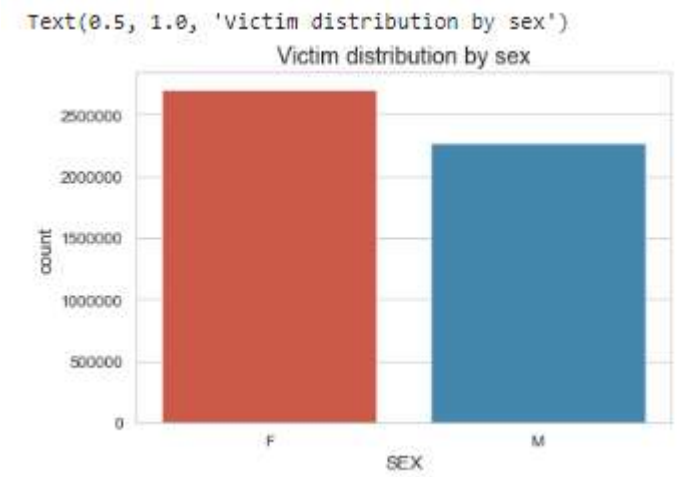



```
x = df[(df['VIC_AGE_GROUP'] !='UNKNOWN')] # filter only person victims not
institutions..
sns.countplot(x=x['VIC_AGE_GROUP'])
plt.ticklabel_format(style='plain', axis='y')
plt.xlabel('Age group')
plt.ylabel('count')
plt.title('Victim distribution by age')
```

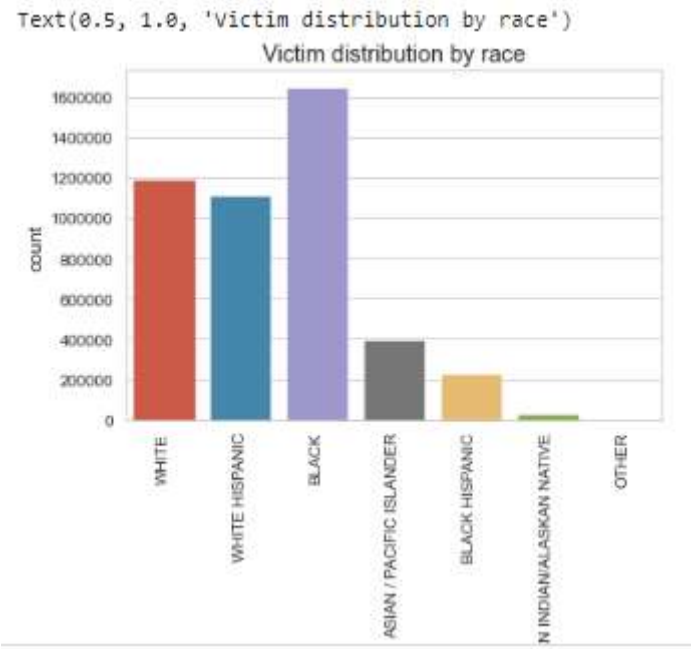
```
Text(0.5, 1.0, 'Victim distribution by age')
```




```
x = df[(df['VIC_SEX']=='M') | (df['VIC_SEX']=='F')] # filter only person
victims not institutions..
sns.countplot(x=x['VIC_SEX'])
plt.ticklabel_format(style='plain', axis='y')
plt.xlabel('SEX')
plt.ylabel('count')
plt.title('Victim distribution by sex')
```




```
x = df[(df['VIC_RACE']!='UNKNOWN')]
sns.countplot(x=x['VIC_RACE'])
plt.ticklabel_format(style='plain', axis='y')
plt.xlabel('RACE')
plt.ylabel('count')
plt.xticks(rotation=90)
plt.title('Victim distribution by race')
```




```
colors = {'felony': '#ff0e0a', 'misdemeanor':'#ff8133', 'violation':
'#ffed47'}

def colorByCrime(crime):
    if crime == 'FELONY':
        return colors['felony']
    elif crime == 'MISDEMEANOR':
        return colors['misdemeanor']
```

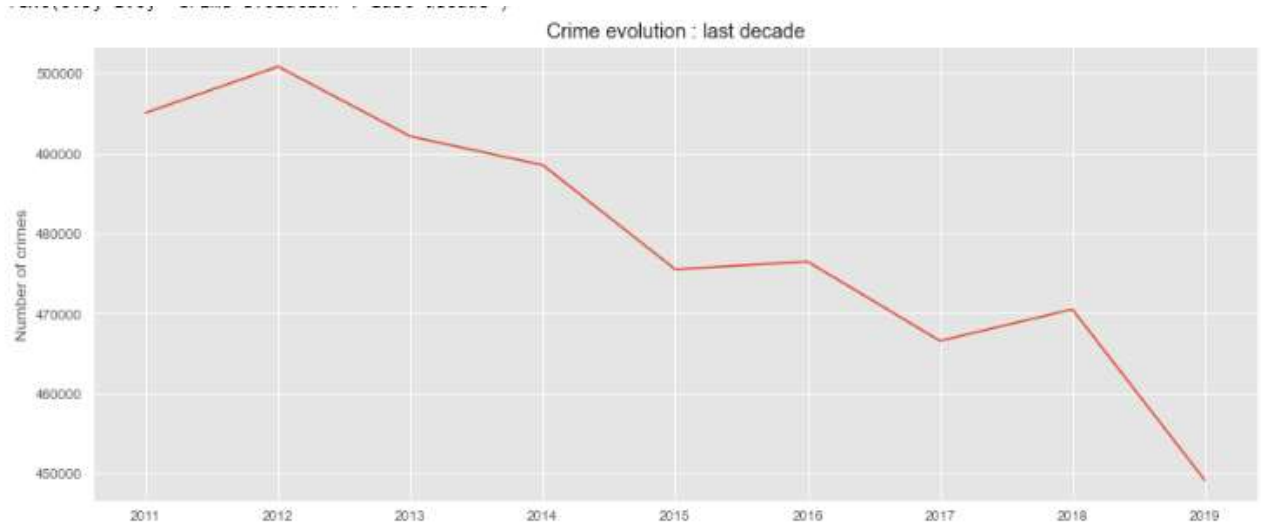
```
    else :
        return colors['violation']

def createLegend():
    template = """
    {% macro html(this, kwargs) %}

    <!doctype html>
    <html lang="en">
    <head>
      <meta charset="utf-8">
    </head>
    <body>
    <div id='maplegend' class='maplegend'
        style='position: absolute; z-index:9999; border:2px solid grey;
background-color:rgba(255, 255, 255, 0.8);
         border-radius:6px; padding: 10px; font-size:14px; right: 20px;
bottom: 20px;'>

    <div class='legend-title'>Legend</div>
    <div class='legend-scale'>
      <ul class='legend-labels'>
        <li><span
style='background:#ff0e0a;opacity:0.7;'></span>felony</li>
        <li><span
style='background:#ff8133;opacity:0.7;'></span>misdemeanor</li>
        <li><span
style='background:#ffed47;opacity:0.7;'></span>violation</li>


</ul>
    </div>
    </div>

    </body>
    </html>

    <style type='text/css'>
      .maplegend .legend-title {
        text-align: left;
        margin-bottom: 5px;
        font-weight: bold;
        font-size: 90%;
```

```
        }
      .maplegend .legend-scale ul {
        margin: 0;
        margin-bottom: 5px;
        padding: 0;
        float: left;
        list-style: none;
        }
      .maplegend .legend-scale ul li {
        font-size: 80%;
        list-style: none;
        margin-left: 0;
        line-height: 18px;
        margin-bottom: 2px;
        }
      .maplegend ul.legend-labels li span {
        display: block;
        float: left;
        height: 16px;
        width: 30px;
        margin-right: 5px;
        margin-left: 0;
        border: 1px solid #999;
        }
      .maplegend .legend-source {
        font-size: 80%;
        color: #777;
        clear: both;
        }
      .maplegend a {
        color: #777;
        }
    </style>
    {% endmacro %}"""
    return template

def generateBaseMap(default_location=[40.704467, -73.892246],
default_zoom_start=13,min_zoom=11,max_zoom=15,):
    base_map = folium.Map(location=default_location, control_scale=True,
zoom_start=default_zoom_start)
    base_map.add_child(ClickForMarker())
    return base_map
def crimeByDate(df, base_map, year, month=0, day=0):
```

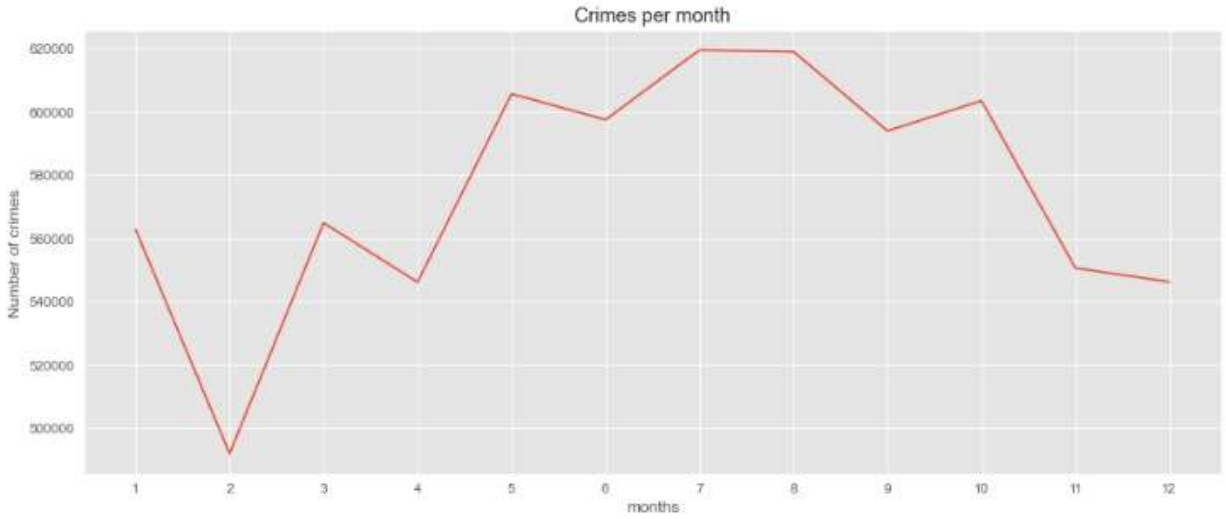
```
    assert year, 'please enter at least a year'
    if (month & day):
        map_df = df[(df['year']== year) & (df['month'] == month) &
(df['day']== day)]
    elif month:
        map_df = df[(df['year']== year) & (df['month'] == month)]
    else :
        map_df = df[(df['year']== year)]

    for index, row in map_df.iterrows():
        color = colorByCrime(row['CRIME_CLASS'])
        folium.CircleMarker([row['Latitude'], row['Longitude']],
                            radius = 3,
                            popup = row['OFNS_DESC'],
                            color = color,
                           ).add_to(base_map)

def heatmapByDate(df, base_map, year, month=0, day=0):
    assert year, 'please enter at least a year'
    if (month & day):
        map_df = df[(df['year']== year) & (df['month'] == month) &
(df['day']== day)]
    elif month:
        map_df = df[(df['year']== year) & (df['month'] == month)]
    else :
        map_df = df[(df['year']== year)]
    dfmatrix = map_df[['Latitude', 'Longitude']].values
    base_map.add_child(plugins.HeatMap(dfmatrix, radius=15))
def transform(row,val_dict,column):
    return val_dict[row[column]]


base_map = generateBaseMap()
crimeByDate(df, base_map, 2018,4,2)
macro = MacroElement()
macro._template = Template(createLegend())
base_map.get_root().add_child(macro)
base_map


new_map = generateBaseMap()
heatmapByDate(df, new_map, 2018,4,2)
new_map
```

```
df2 =
df[df['BORO_NM']!='UNKNOWN'].groupby(['BORO_NM'])['CMPLNT_NUM'].count().res
et_index()
data =
df[df['BORO_NM']!='UNKNOWN'].groupby(['BORO_NM'])['CMPLNT_NUM'].size()
palette = sns.color_palette("magma",5)
rank = data.argsort().argsort()
g=sns.barplot(x='BORO_NM',y='CMPLNT_NUM',data=df2,palette=np.array(palette[
::-1])[rank]);
plt.xlabel('Borough')
plt.ylabel('Number of crimes');
plt.ticklabel_format(style='plain', axis='y')
plt.title("Number of crimes per Borough");


borough_area = {'BROOKLYN':179.7, 'STATEN ISLAND':148.9, 'BRONX':109.3,
'QUEENS':281.5, 'MANHATTAN':58.8}
df2 =
df[df['BORO_NM']!='UNKNOWN'].groupby(['BORO_NM'])['CMPLNT_NUM'].count().res
et_index()
df2['Area'] = df2.apply(transform, val_dict=borough_area, column='BORO_NM',
axis=1);
df2['CrimeDensityArea'] = df2.CMPLNT_NUM / df2.Area
df2.head()
data = df2['CrimeDensityArea']
palette = sns.color_palette("magma",5)
rank = data.argsort().argsort()
sns.set_style('whitegrid');
g=sns.barplot(x='BORO_NM',y='CrimeDensityArea',data=df2,palette=np.array(pa
lette[::-1])[rank]);
plt.ylabel('Number of Crimes/area');
plt.title("Crime Density per Borough by Area");


borough_pop_18 = {'BROOKLYN':2582830, 'STATEN ISLAND':476179,
'BRONX':1432130, 'QUEENS':2278910, 'MANHATTAN':1628700}
df3 = df[(df['BORO_NM']!='UNKNOWN') &
(df['year']==2018)].groupby(['BORO_NM'])['CMPLNT_NUM'].count().reset_index(
)
df3['Population18'] = df3.apply(transform, val_dict=borough_area,
column='BORO_NM', axis=1);
df3['CrimeDensityPop'] = df3.CMPLNT_NUM / df3.Population18
data = df3['CrimeDensityPop']
```

```
palette = sns.color_palette("magma",5)
rank = data.argsort().argsort()
sns.set_style('whitegrid');
g=sns.barplot(x='BORO_NM',y='CrimeDensityPop',data=df3,palette=np.array(pal
ette[::-1])[rank]);
plt.ylabel('Number of Crimes/population');
plt.title("Crime Density per Borough by Population 2018");


pct_cutoff=5
fig= plt.figure(figsize=(15,6))

the predefined cutoff value
def my_autopct(pct):
    return ('%1.0f%%' % pct) if pct > pct_cutoff else ''

df_temp=df['PREM_TYP_DESC'].value_counts(normalize=True).round(8)

labels = [n if v > pct_cutoff/100 else ''
          for n, v in zip(df_temp.index, df_temp)]

plt.pie(df_temp, labels=labels, autopct=my_autopct, shadow=False)

plt.title('Crime occurance per premise')

plt.show()


fig= plt.figure(figsize=(15,6))
temp_df = df[df["year"]>2010]
temp_df.groupby('year').count()["CMPLNT_NUM"].plot(kind='line')
plt.xlabel('')
plt.ylabel('Number of crimes')
plt.title('Crime evolution : last decade')
```

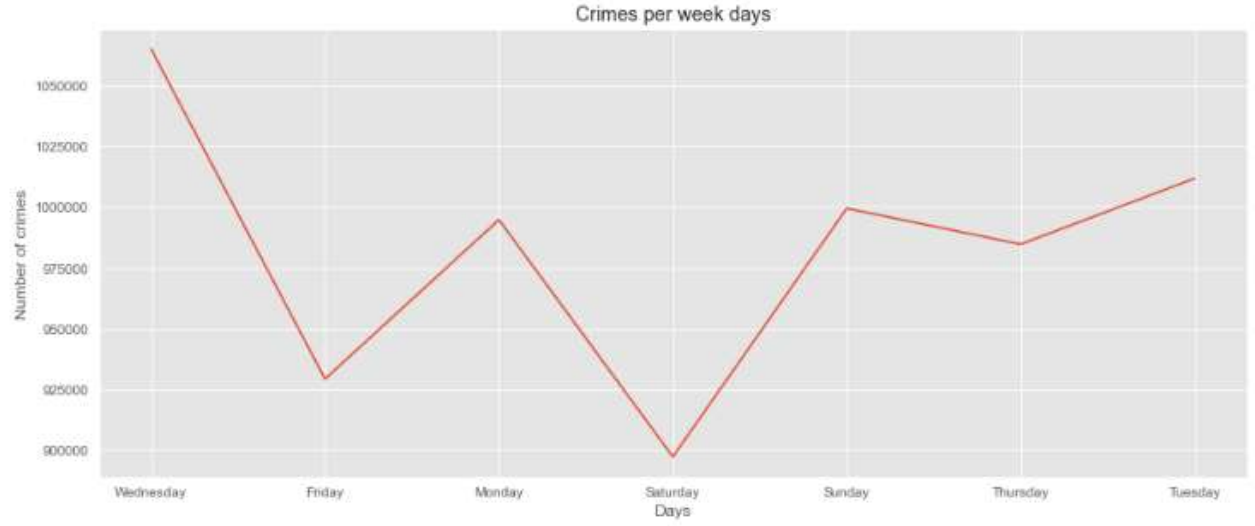



```
fig= plt.figure(figsize=(15,6))
df.groupby('month').count()["CMPLNT_NUM"].plot(kind='line')
plt.xlabel('months')
plt.xticks([x for x in range(1,13)])
plt.ylabel('Number of crimes')
plt.title('Crimes per month')
```

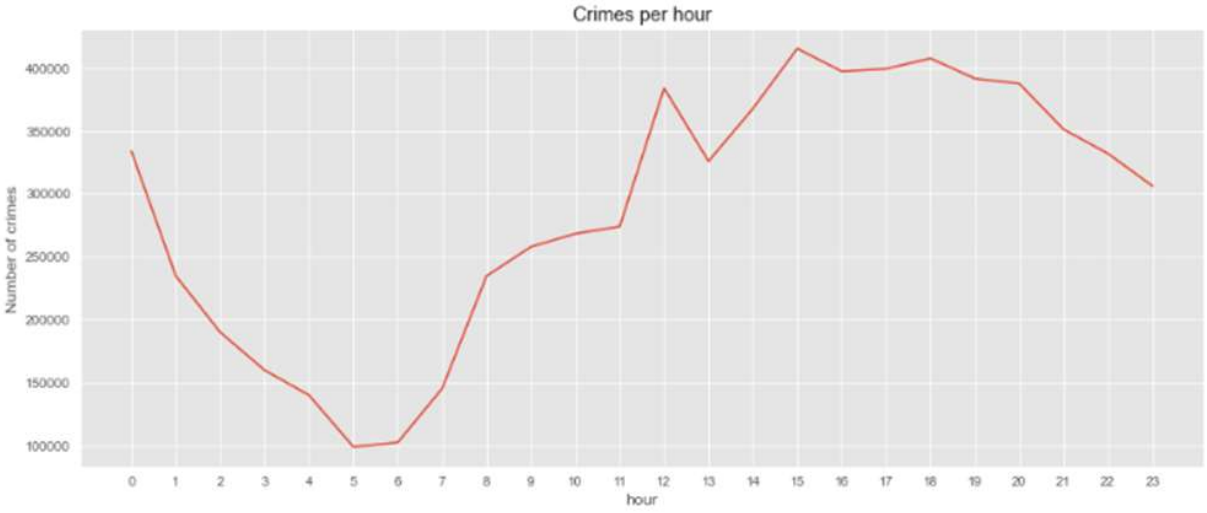



```
fig= plt.figure(figsize=(15,6))
df.groupby('weekday').count()["CMPLNT_NUM"].plot(kind='line')
plt.xlabel('Days')
plt.xticks([x for x in range(7)])
plt.ylabel('Number of crimes')
plt.ticklabel_format(style='plain', axis='y')
plt.title('Crimes per week days')
```

```
Text(0.5, 1.0, 'Crimes per week days')
```




```
fig= plt.figure(figsize=(15,6))
df.groupby('weekday').count()["CMPLNT_NUM"].plot(kind='line')
plt.xlabel('hour')
plt.xticks([x for x in range(24)])
plt.ylabel('Number of crimes')
plt.title('Crimes per hour')
```




```
df.head()
```

	CMPLNT_NUM	year	month	day	weekday	hour	Latitude	Longitude	CRM_ATPT_CPTD_CD	OFNS_DESC	...	PREM_TYP_DESC	IN_PARK	IN_PUBLIC_HOUSING	IN_STATION	SUSP_AGE_GROUP	SUSP_RACE	SUSP_SEX	VIC_AGE_
0	724718389	2014	9	4	Thursday	17	40.685041	-73.921777	COMPLETED	ASSAULT 3 & RELATED OFFENSES	...	STREET	0	0	0	UNKNOWN	UNKNOWN	U	
1	191133903	2016	10	12	Wednesday	7	40.636991	-74.134093	COMPLETED	GRAND LARCENY	...	STREET	0	0	0	UNKNOWN	BLACK	U	
2	720151206	2012	9	28	Friday	13	40.823876	-73.891863	COMPLETED	GRAND LARCENY	...	STREET	0	0	0	UNKNOWN	WHITE HISPANIC	M	
3	232242098	2015	3	24	Tuesday	15	40.845707	-73.910398	COMPLETED	PETIT LARCENY	...	STREET	0	0	0	UNKNOWN	BLACK	M	
4	708078702	2017	5	20	Saturday	4	40.763992	-73.828426	COMPLETED	ASSAULT 3 & RELATED OFFENSES	...	STREET	0	0	0	25-44	WHITE HISPANIC	M	

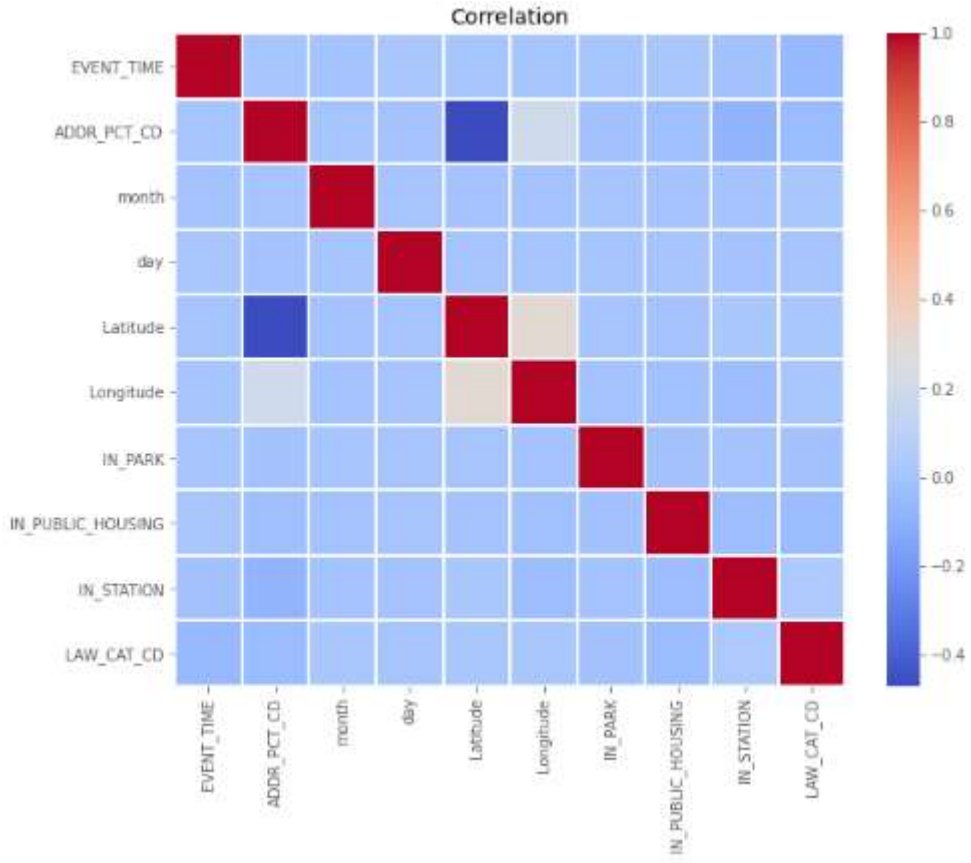
5 rows × 23 columns

2.ipynb

```
import numpy as np
import pandas as pd
import joblib
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
pd.set_option('display.max_rows', 200)
pd.set_option('display.max_columns', 200)
plt.style.use('ggplot')

from sklearn.ensemble import RandomForestClassifier
from xgboost import XGBClassifier
from sklearn.model_selection import train_test_split
from sklearn.model_selection import GridSearchCV, RandomizedSearchCV
from sklearn.feature_selection import SelectFromModel
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
from sklearn.metrics import roc_curve, auc
import scikitplot as skplt

df.LAW_CAT_CD.value_counts().sort_values(ascending=False)

zero,one,two = [],[],[]
zero_c,one_c,two_c = 0,0,0
for i in tqdm(df.iterrows()):
    if i[1].LAW_CAT_CD == 0 and zero_c <= 817326:
        zero.append(i[1].values)
        zero_c += 1
    elif i[1].LAW_CAT_CD == 1 and one_c <= 817326:
        one.append(i[1].values)
        one_c += 1
    elif i[1].LAW_CAT_CD == 2 and two_c <= 817326:
        two.append(i[1].values)
        two_c += 1
    if zero_c == 817326 and one_c == 817326 and two_c == 817326:
        break
zero_df = pd.DataFrame(zero, columns=df.columns.values.tolist())
one_df = pd.DataFrame(one, columns=df.columns.values.tolist())
two_df = pd.DataFrame(two, columns=df.columns.values.tolist())
final_df = pd.concat([zero_df,one_df,two_df])
```

```
final_df.to_csv("./ny_clean_train_balanced.csv",index=False)

df.info()
df.head()

def reduce_mem_usage(df, verbose=True):
    numerics = ['int16', 'int32', 'int64', 'float16', 'float32', 'float64']
    start_mem = df.memory_usage().sum() / 1024**2
    for col in df.columns:
        col_type = df[col].dtypes
        if col_type in numerics:
            c_min = df[col].min()
            c_max = df[col].max()
            if str(col_type)[:3] == 'int':
                if c_min > np.iinfo(np.int8).min and c_max <
np.iinfo(np.int8).max:
                    df[col] = df[col].astype(np.int8)
                elif c_min > np.iinfo(np.int16).min and c_max <
np.iinfo(np.int16).max:
                    df[col] = df[col].astype(np.int16)
                elif c_min > np.iinfo(np.int32).min and c_max <
np.iinfo(np.int32).max:
                    df[col] = df[col].astype(np.int32)
                elif c_min > np.iinfo(np.int64).min and c_max <
np.iinfo(np.int64).max:
                    df[col] = df[col].astype(np.int64)
            else:
                if c_min > np.finfo(np.float16).min and c_max <
np.finfo(np.float16).max:
                    df[col] = df[col].astype(np.float16)
                elif c_min > np.finfo(np.float32).min and c_max <
np.finfo(np.float32).max:
                    df[col] = df[col].astype(np.float32)
                else:
                    df[col] = df[col].astype(np.float64)
    end_mem = df.memory_usage().sum() / 1024**2
    print('Memory usage after optimization is: {:.2f} MB'.format(end_mem))
    print('Decreased by {:.1f}%'.format(100 * (start_mem - end_mem) /
start_mem))
    return df
df = reduce_mem_usage(df)

feature_lst=['EVENT_TIME','ADDR_PCT_CD', 'month', 'day', 'Latitude',
```

```
        'Longitude', 'BORO_NM',"WEEKDAY",
        'IN_PARK', 'IN_PUBLIC_HOUSING', 'IN_STATION', 'VIC_AGE_GROUP',
        'VIC_RACE', 'VIC_SEX','LAW_CAT_CD']

df_sel=df[feature_lst].copy()
df_sel.info()
df_sel.head()

print(df_sel.shape)
df_sel.LAW_CAT_CD.value_counts().sort_values(ascending=False)
(2451980, 14)
1    817327
2    817327
0    817326

corr = df_sel.corr()
plt.figure(figsize = (10,8))
sns.heatmap(corr, cmap = "coolwarm", linewidth = 2, linecolor = "white")
plt.title("Correlation")
plt.show()
```

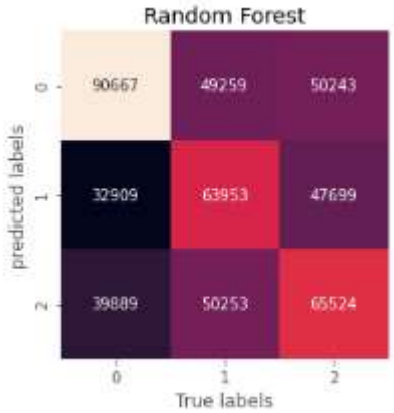



```
df_state_dummy = pd.get_dummies(df_sel)
df_state_dummy.info()
```

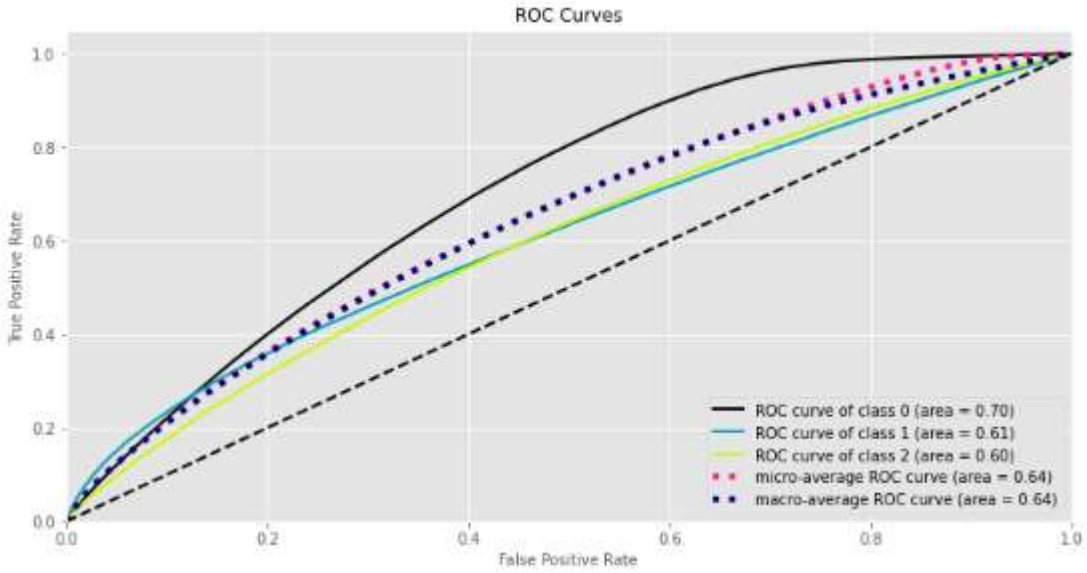
```
df=df_state_dummy
target='LAW_CAT_CD'
y = df[target]
y.unique()
y.value_counts()
1    817327
2    817327
0    817326

X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2,shuffle=True, random_state=21, stratify=y)

def plot_cm(y_pred,y_test,algorithm,figure_name):
    mat_RF = confusion_matrix(y_pred,y_test)
    plt.figure(figsize=(16,4))
    sns.heatmap(mat_RF, square=True, annot=True, fmt='d',
cbar=False,xticklabels=[0,1,2],yticklabels=[0,1,2])
    plt.xlabel('True labels')
    plt.ylabel('predicted labels')
    plt.title(algorithm)
    plt.savefig(figure_name)

def plot_roc(y_test, model, figure_name):
    pl = skplt.metrics.plot_roc(y_test, model.predict_proba(X_test),
figsize=(12,6))
    plt.show()
    pl.figure.savefig(figure_name)

def save_model(model, model_name,is_tree=False):
    joblib.dump(model.estimators_[0] if is_tree else
model,f'{model_name}.joblib')
    print(f"Model size: {np.round(os.path.getsize(f'{model_name}.joblib') /
1024 / 1024, 2) } MB")
clf=RandomForestClassifier(n_estimators=100,n_jobs=-1,verbose=1)
clf.fit(X_train,y_train)
y_pred=clf.predict(X_test)
acc_rf = accuracy_score(y_test, y_pred)
class_report = classification_report(y_test, y_pred)
print(class_report)
```

```
precision    recall  f1-score   support

           0       0.48      0.55      0.51    163465
           1       0.44      0.39      0.42    163465
           2       0.42      0.40      0.41    163466

    accuracy                           0.45    490396
   macro avg       0.45      0.45      0.45    490396
weighted avg       0.45      0.45      0.45    490396
```

```
plot_cm(y_pred,y_test,"Random Forest")
```




```
plot_roc(y_test,clf)
```




```
save_model(clf,"random_forest",True)
```

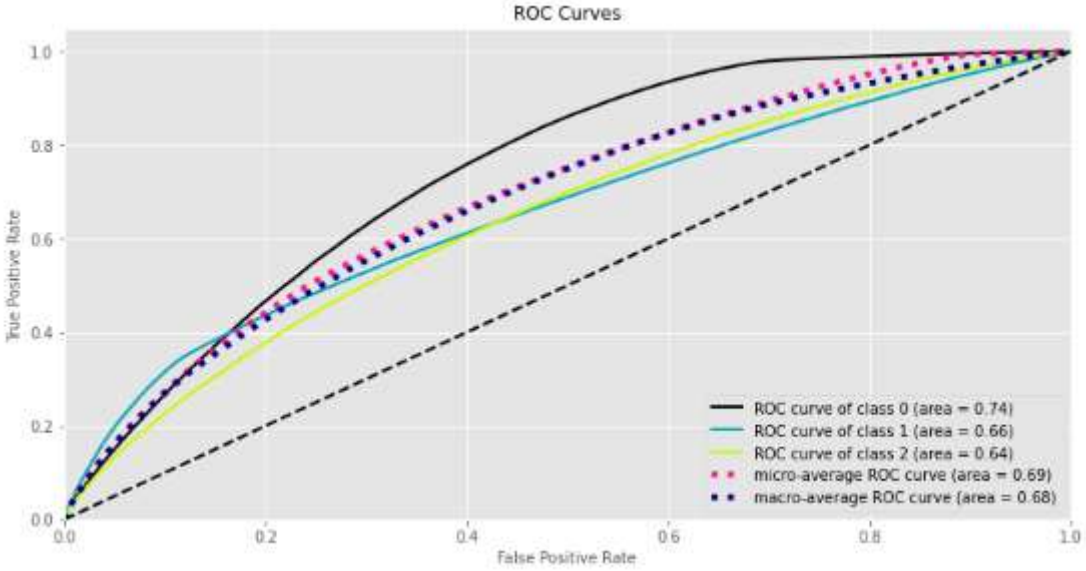
```
from the columns name
regex = re.compile(r"\[|\]|<", re.IGNORECASE)
X_train.columns = [regex.sub("_", col) if any(x in str(col) for x in
set(('[', ']', '<'))) else col for col in X_train.columns.values]
```

```
params = {
            'objective':'multi:softmax',
            'max_depth': 10,
            'alpha': 10,
            'learning_rate': 0.1,
            'n_estimators':100,
            'use_label_encoder':False
        }
xgb_clf = XGBClassifier(**params)
xgb_clf.fit(X_train, y_train)

y_pred = xgb_clf.predict(X_test)
accuracy = accuracy_score(y_pred, y_test)
print('XGBBoost Model accuracy score:
{0:0.4f}'.format(accuracy_score(y_test, y_pred)))
class_report = classification_report(y_test, y_pred)
print(class_report)
```

	precision	recall	f1-score	support
0	0.49	0.76	0.60	163465
1	0.59	0.35	0.44	163465
2	0.47	0.40	0.43	163466
accuracy			0.50	490396
macro avg	0.52	0.50	0.49	490396
weighted avg	0.52	0.50	0.49	490396

```
plot_cm(y_pred, y_test, "XGBBoost")
```


```
plot_roc(y_test, lbm_clf)
```


ROC Curves
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0
ROC curve of class 0 (area = 0.74)
ROC curve of class 1 (area = 0.66)
ROC curve of class 2 (area = 0.64)
micro-average ROC curve (area = 0.69)
macro-average ROC curve (area = 0.68)