

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

**ĐỒ ÁN: PHÂN TÍCH**

**CÁC THUẬT TOÁN KHAI THÁC**

MÔN: KHAI THÁC DỮ LIỆU

*GVHD:* ***Nguyễn Hồ Duy Trí***

|  |  |
| --- | --- |
| **HỌ & TÊN** | **MSSV** |
| Hà Trọng Nhân | 17520836 |
| Nguyễn Thị Ngọc Hà | 17520421 |
| Phan Thị Long Huệ | 17520530 |
| Khổng Minh Quốc | 17520954 |

# LỜI NÓI ĐẦU

Ngày này, Trí Tuệ Nhân Tạo đang len lỏi vào mọi lĩnh vực trong đời sống như một bằng chứng của cuộc cách mạng công nghiệp lần thứ tư và ngày càng khẳng định vai trò to lớn của nó trong xã hội ngày một phát triển hiện nay. Machine Learning - một lĩnh vực con của AI cũng đang trở nên quan trọng hơn trong ngành Công nghệ thông tin với những tiện ích mà nó đem lại.

Theo Wikipedia, ***Machine learning is the subfield of computer science that “gives computers the ability to learn without being explicitly programmed”*.** Nói đơn giản, Machine Learning là một lĩnh vực nhỏ của Khoa Học Máy Tính, nó có khả năng tự học hỏi dựa trên dữ liệu đưa vào mà không cần phải được lập trình cụ thể

Machine Learning đã giúp con người rất nhiều trong việc phân loại và dự đoán dữ liệu. Các ứng dụng quen thuộc của Machine Learning đối với con người có thể ví như: Hệ thống gợi ý sản phẩm, gợi ý phim; Hệ thống dự báo thời tiết,…

Những năm gần đây, khi mà khả năng tính toán của các máy tính được nâng lên một tầm cao mới, xu hướng phát triển Công nghệ thông tin ngày càng tăng, song song với nó là lượng dữ liệu được thu thập ngày một trở nên khổng lồ. Vì vậy, nhu cầu để xử lý dữ liệu cũng lớn hơn. Machine Learning chính là giải pháp tối ưu và không thể thiếu để giải quyết điều đó.

# NHẬN XÉT BÁO CÁO

MỤC LỤC

[LỜI NÓI ĐẦU 2](#_Toc43963873)

[NHẬN XÉT BÁO CÁO 3](#_Toc43963874)

[I. GIỚI THIỆU 5](#_Toc43963875)

[1. Về Bộ dữ liệu: 5](#_Toc43963876)

[2. Lý do chọn đề tài: 5](#_Toc43963877)

[3. Mô tả bộ dữ liệu: 6](#_Toc43963878)

[II. BÀI TOÁN VÀ TIỀN XỬ LÝ DỮ LIỆU: 13](#_Toc43963879)

[1. Bài toán: 13](#_Toc43963880)

[2. Tiền xử lý dữ liệu: 26](#_Toc43963881)

[III. THUẬT TOÁN: 41](#_Toc43963882)

[1. Lý do lựa chọn: 41](#_Toc43963883)

[2. Các thuật toán: 45](#_Toc43963884)

[2.1. Random Forest: 45](#_Toc43963885)

[2.2. Neural Network: 54](#_Toc43963886)

[2.3. K – Nearest Neighbors: 58](#_Toc43963887)

[2.4. Ensemble Voting: 62](#_Toc43963888)

[IV. KẾT QUẢ: 67](#_Toc43963889)

[1. So sánh: 67](#_Toc43963890)

[2. Đánh giá: 68](#_Toc43963891)

[V. KẾT LUẬN: 70](#_Toc43963892)

[1. Ưu điểm: 70](#_Toc43963893)

[2. Hạn chế: 71](#_Toc43963894)

[3. Hướng phát triển: 72](#_Toc43963895)

[BẢNG PHÂN CÔNG: 75](#_Toc43963896)

[BẢNG ĐÁNH GIÁ CHÉO: 76](#_Toc43963897)

[TÀI LIỆU THAM KHẢO: 77](#_Toc43963898)

# GIỚI THIỆU

### Về Bộ dữ liệu:

Với khả năng ưu việt trong việc phân loại và dự đoán dữ liệu, Machine Learning đã giúp con người đưa ra những dự đoán tương đối để hoạt động trong nhiều ngành công nghiệp hiện nay.

Phải nói đến là ứng dụng dự báo thời tiết vô cùng thiết thực đối với con người. Song song với đó, nhóm chúng em cũng muốn dùng Machine Learning để đưa ra phân tích về các vấn đề thiết thực khác trong xã hội – dữ liệu tội phạm ở Chicago chính là bộ dữ liệu mà nhóm hướng tới.

Đường dẫn bộ dữ liệu: <https://bitly.com.vn/95Dx8>

### Lý do chọn đề tài:

Ngày nay trên phạm vi toàn cầu với sự bùng nổ của khoa học công nghệ, sự phát triển không ngừng của kinh tế, thương mại và mức sống của người dân ngày càng được nâng cao đã gắn kết hoạt động du lịch, giải trí, vận dụng công nghệ của con người với công ăn việc làm. Hàng năm, những sự kiện chính trị, kinh tế, văn hoá, thể thao, công nghệ mang tầm quốc gia, khu vực và quốc tế diễn ra với mật độ dày đặc đã trở thành một cơ hội tốt cho mỗi quốc gia phát triển. Đời sống con người được cải thiện, tuổi thọ cũng được tăng cao nhờ sự phát triển của y tế, nhu cầu người dân mỗi lúc được tăng cao và sự đảm bảo an toàn đã trở nên cấp thiết hơn bao giờ hết.

Bên cạnh sự phát triển của mỗi quốc gia, đều tồn tại những thành phần chống đối, làm kìm hãm sự phát triển cả về xã hội lẫn kinh tế.Nhận thấy sự cần thiết của việc “phải loại trừ” những thành phần gây cản trở cho kinh tế - xã hội, cản trở cho sự phát triển của đất nước, vì thế mỗi quốc gia đã lên chiến lược phân tích các hành vi của những kẻ cản trở - tội phạm này một cách tỉ mỉ, chính xác nhằm giảm thiểu tối đa sự chống đối cũng như đảm bảo sự an toàn cho người dân.

Hình thức gây án của mỗi tên tội phạm, đều rất đa dạng và khó kiểm soát. Vì thế việc xác định ngày xảy ra vụ án, vị trí xảy ra vụ án, có liên quan đến ai, nơi đó có thường xuyên xảy ra không, vv là điều vô cùng quan trọng.

Nhận thấy đề tài Phân tích tội phạm tại Thành phố Chicago từ năm 2001 đến năm 2017 là một đề tài bổ ích và mang tính cấp thiết, nó không những giúp bọn em hiểu sâu hơn về thuật toán mà còn giúp mở rộng tư duy nghiên cứu. Vì thế nhóm em quyết định chọn đề tài làm báo cáo.

### Mô tả bộ dữ liệu:

Bộ dữ liệu phản ánh các sự cố tội phạm được báo cáo ( ngoại trừ các vụ án giết người có dữ liệu cho từng nạn nhân ) xảy ra ở Thành phố Chicago từ năm 2001 đến năm 2017.

Dữ liệu được trích xuất từ ​​hệ thống Báo cáo và Phân tích Thi hành Luật Công dân của Sở Cảnh sát Chicago. Để bảo vệ quyền riêng tư của nạn nhân, tội phạm, địa chỉ chỉ được hiển thị ở cấp độ khối và các địa điểm cụ thể không được xác định. Bộ dữ liệu luôn được cập nhật thường xuyên, chính vì vậy mà việc phân loại tội phạm sơ bộ có thể được thay đổi dựa theo các điều tra bổ sung và luôn có khả năng xảy ra lỗi cơ học và con người.

Do đó, thông tin không nên sử dụng cho mục đích so sánh theo thời gian vì không đảm bảo được tính chính xác, đầy đủ, kịp thời hoặc trình tự chính xác của thông tin.

Dựa vào bộ dữ liệu trên, chúng ta sẽ sử dụng các thuật toán trong Khai thác dữ liệu nhằm mục đích phân loại dự đoán của các loại tội phạm ở Thành phố Chicago.

Dữ liệu gồm có:

* Thuộc tính: 23 (cột)
* Số dòng dữ liệu: 7941282
* Bảng mô tả dữ liệu

|  |  |  |
| --- | --- | --- |
| **Tên cột** | **Mô tả** | **Loại** |
| **#** | Không xác định. | Int |
| **ID** | Định danh duy nhất cho hồ sơ. | Int |
| **Case Number** | Số RD (Records Division ) của Sở Cảnh sát Chicago, là thông tin duy nhất cho vụ án. | Object |
| **Date** | Ngày xảy ra vụ án. Điều này đôi khi là một ước tính tốt nhất. | Object |
| **Block** | Địa chỉ đã được xử lý lại một phần tại nơi xảy ra vụ án, đặt nó trên cùng một khối với địa chỉ thực tế. | Object |
| **IUCR** | Mã báo cáo tội phạm thống nhất Illinois (IUCR) là bốn mã số mà các cơ quan thực thi pháp luật sử dụng để phân loại các vụ việc hình sự khi lấy báo cáo cá nhân. Các mã này cũng được sử dụng để tổng hợp các loại trường hợp cho mục đích thống kê. Mã này được liên kết trực tiếp với Primary Type và Description. | Object |
| **Primary Type** | Mô tả chính của mã IUCR. | Object (Categorial):   * Theft * Battery * Criminal damage * Narcotics * Other offense * Burglary * Assault * Motor vehicle theft * Robbery * Deceptive practice * Criminal trespass * Prostitution * Weapons violation * Public peace violation * Offense involving children * Crim sexual assault * Sex offense * Gambling * Liquor law violation * Interference with public officer * Arson * Homicide * Kidnapping * Intimidation * Stalking * Obscenity * Public indecency * Other narcotic violation |
| **Description** | Mô tả phụ của mã IUCR, là một danh mục con của mô tả chính. | Object |
| **Location Description** | Mô tả vị trí xảy ra vụ án. | Object |
| **Arrest** | Cho biết liệu một vụ án đó đã được bắt giữ chưa. | Bool:   * True * False |
| **Domestic** | Cho biết liệu vụ án đó có liên quan đến gia đình theo quy định của Luật Bạo hành Gia đình ở Illinois hay không. | Bool:   * True * False |
| **Beat**(Tạm dịch là khu vực đi tuần của cảnh sát) | Cho biết khu vực đi tuần nơi xảy ra vụ án. Một khu vực đi tuần chính là khu vực địa lý cảnh sát nhỏ nhất - mỗi khu vực đi tuần có một xe cảnh sát chuyên dụng. Ba đến năm khu vực đi tuần tạo thành một khu vực cảnh sát, và ba khu vực tạo thành một khu cảnh sát. Sở cảnh sát Chicago có 22 khu cảnh sát. | Int |
| **District** | Cho biết khu vực cảnh sát nơi xảy ra vụ án | Float |
| **Ward** | Phường- nơi xảy ra vụ án. | Float |
| **Community Area** | Khu vực cộng đồng - Cho biết khu vực cộng đồng nơi xảy ra vụ án. Chicago có 77 khu vực cộng đồng. | Float |
| **FBI Code** | Bộ luật FBI - Biểu thị phân loại tội phạm như được nêu trong Hệ thống báo cáo sự cố quốc gia (NIBRS) của FBI. | Object |
| **X Coordinate** | Tọa độ X - Tọa độ x của vị trí xảy ra vụ án trong phép chiếu State Plane tại Bang Illinois phía đông NAD 1983. Vị trí này được chuyển từ vị trí thực tế đã được điều chỉnh một phần nhưng rơi vào cùng một khối. | Float |
| **Y Coordinate** | Tọa độ Y - Tọa độ y của vị trí xảy ra vụ án trong phép chiếu State Plane tại Bang Illinois phía đông NAD 1983. Vị trí này được chuyển từ vị trí thực tế đã được điều chỉnh một phần nhưng rơi vào cùng một khối. | Object |
| **Year** | Năm xảy ra vụ án. | Float |
| **Updated On** | Ngày và thời gian bản ghi án được cập nhật lần cuối. | Object |
| **Latitude** | Vĩ độ của vị trí xảy ra vụ án. Vị trí này được chuyển từ vị trí thực tế để điều chỉnh một phần nhưng rơi vào cùng một khối. | Object |
| **Longitude** | Kinh độ của địa điểm xảy ra vụ án. Vị trí này được chuyển từ vị trí thực tế để điều chỉnh một phần nhưng rơi vào cùng một khối. | Float |
| **Location** | Vị trí - Vị trí xảy ra vụ án ở định dạng cho phép tạo bản đồ và các hoạt động địa lý khác trên cổng dữ liệu này. Vị trí này được chuyển từ vị trí thực tế để điều chỉnh một phần nhưng rơi vào cùng một khối. | Object |

# BÀI TOÁN VÀ TIỀN XỬ LÝ DỮ LIỆU:

### Bài toán:

Dự đoán nhãn của các vụ án, mà cụ thể ở đây là loại tội phạm ở Chicago.

* Đầu vào:

Toàn bộ dữ liệu gồm 23 thuộc tính trong đó có 7941282 dòng dữ liệu

|  |  |
| --- | --- |
| **Tên cột** | **Loại** |
| **#** | Int |
| **ID** | Int |
| **Case Number** | Object |
| **Date** | Object |
| **Block** | Object |
| **IUCR** | Object |
| **Primary Type** | Object (Categorial):   * Theft * Battery * Criminal damage * Narcotics * Other offense * Burglary * Assault * Motor vehicle theft * Robbery * Deceptive practice * Criminal trespass * Prostitution * Weapons violation * Public peace violation * Offense involving children * Crim sexual assault * Sex offense * Gambling * Liquor law violation * Interference with public officer * Arson * Homicide * Kidnapping * Intimidation * Stalking * Obscenity * Public indecency * Other narcotic violation |
| **Description** | Object |
| **Location Description** | Object |
| **Arrest** | Bool:   * True * False |
| **Domestic** | Bool:   * True * False |
| **Beat**(Tạm dịch là khu vực đi tuần của cảnh sát) | Int |
| **District** | Float |
| **Ward** | Float |
| **Community Area** | Float |
| **FBI Code** | Object |
| **X Coordinate** | Float |
| **Y Coordinate** | Object |
| **Year** | Float |
| **Updated On** | Object |
| **Latitude** | Object |
| **Longitude** | Float |
| **Location** | Object |

* Đầu ra:

|  |  |
| --- | --- |
| **Tên cột** | **Loại** |
| **Primary Type** | Object (Categorial):   * Theft * Battery * Criminal damage * Narcotics * Other offense * Burglary * Assault * Motor vehicle theft * Robbery * Deceptive practice * Criminal trespass * Prostitution * Weapons violation * Public peace violation * Offense involving children * Crim sexual assault * Sex offense * Gambling * Liquor law violation * Interference with public officer * Arson * Homicide * Kidnapping * Intimidation * Stalking * Obscenity * Public indecency * Other narcotic violation |

* Thuật toán được sử dụng:
* **Thuật toán Random Forest**

*Random Forest là gì?*

Như tên gọi của nó Random Forest — rừng ngẫu nhiên: đây là phương pháp xây dựng một tập hợp rất nhiều cây quyết định và sử dụng phương pháp voting để đưa ra quyết định về biến target cần được dự báo.

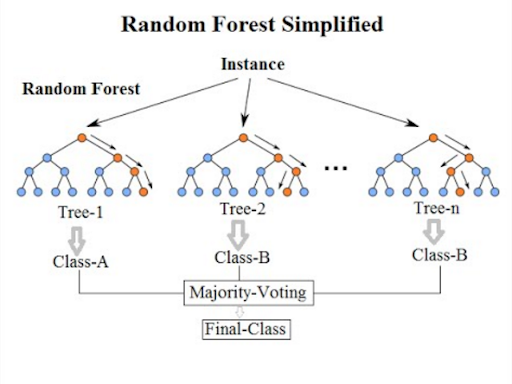
Về mặt toán học thuật toán có thể được giải thích như sau: Random Forest là một tập hợp của hàng trăm Decision Tree, trong đó mỗi Decision Tree được tạo nên ngẫu nhiên từ việc tái chọn mẫu (chọn random 1 phần của data để xây dựng) và random các biến từ toàn bộ các biến trong trong data. Ngoài ra tập mẫu của mỗi cây cũng được lựa chọn ngẫu nhiên bằng phương pháp Bootstrap từ tập mẫu ban đầu. Với một cơ chế như vậy, Random Forest cho ta một kết quả chính xác rất cao nhưng đánh đổi bằng việc ta không thể hiểu cơ chế hoạt động của thuật toán này do cấu trúc quá phức tạp của mô hình này — do vậy thuật toán này là một trong những phương thức Black Box — tức ta sẽ bỏ tay vào bên trong và rút ra được kết quả chứ không thể giải thích được cơ chế hoạt động của mô hình.

Random Forest là một phương pháp Supervised Learning do vậy có thể xử lý được các bài toán về Classification (phân loại) và Regression (dự báo về các giá trị).

*Decision Tree là gì ?*

Decision Tree là tên đại diện cho một nhóm thuật toán phát triển dựa trên Cây quyết định. Ở đó, mỗi Node của cây sẽ là các thuộc tính, và các nhánh là giá trị lựa chọn của thuộc tính đó. Bằng cách đi theo các giá trị thuộc tính trên cây, cây quyết định sẽ cho ta biết giá trị dự đoán.

*Mô hình Random Forest:*



Hình 1. Mô hình thuật toán Random Forest

* **Thuật toán Neural Network**:

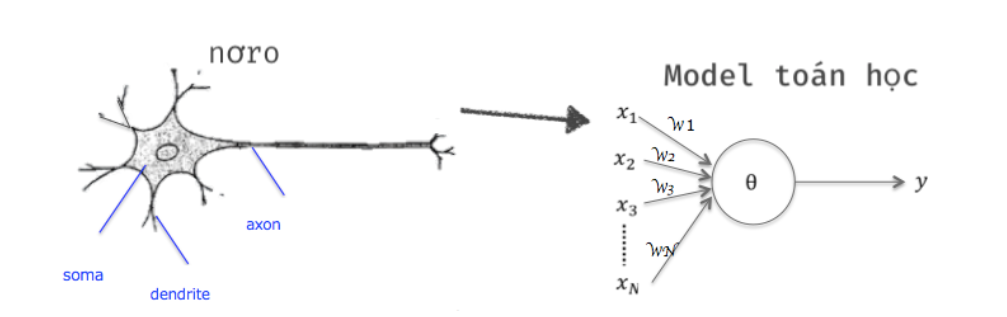
*Khái niệm Neural Network:*

Mạng nơ-ron nhân tạo là một chuỗi các [thuật toán](https://vietnambiz.vn/giao-dich-thuat-toan-algorithmic-trading-la-gi-nhung-dac-diem-can-luu-y-20191205025413985.htm)được đưa ra để nỗ lực tìm kiếm các [mối quan hệ](https://vietnambiz.vn/tuong-quan-dong-bien-positive-correlation-la-gi-nhung-dac-diem-can-luu-y-20191227113029107.htm) cơ bản trong một tập hợp dữ liệu, thông qua quá trình bắt chước cách thức hoạt động của bộ não con người.

Theo nghĩa này, mạng nơ-ron nhân tạo được xem như hệ thống các tế bào thần kinh nhân tạo, có thể là hữu cơ hoặc nhân tạo về bản chất.

Mạng nơ-ron nhân tạo có thể thích ứng với các thay đổi trong [đầu vào](https://vietnambiz.vn/phan-tich-dau-vao-dau-ra-input-output-analysis-la-gi-20191203170851464.htm), do đó, nó đưa ra các kết quả tốt nhất có thể mà không cần phải thiết kế lại các tiêu chí đầu ra.

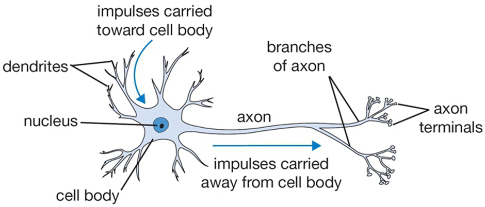
Khái niệm mạng nơ-ron nhân tạo có nguồn gốc từ [trí tuệ nhân tạo](https://vietnambiz.vn/tri-tue-nhan-tao-artificial-intelligence-ai-la-gi-20190924104039225.htm), đang nhanh chóng trở nên phổ biến trong sự phát triển của các [hệ thống giao dịch điện tử](https://vietnambiz.vn/giao-dich-dien-tu-electronic-transaction-la-gi-20200110110821984.htm).



Hình 2. Mô hình thuật toán Neural Network

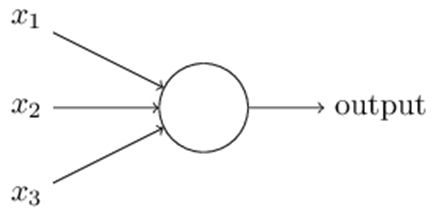
* Perceptron cơ bản

Một mạng nơ-ron được cấu thành bởi các nơ-ron đơn lẻ được gọi là các *perceptron*. Nơ-ron nhân tạo được lấy cảm hứng từ nơ-ron sinh học như hình mô tả bên dưới:



Hình 3. Mô hình mạng Nơ ron

Như hình trên, ta có thể thấy một nơ-ron có thể nhận nhiều đầu vào và cho ra một kết quả duy nhất. Mô hình của perceptron cũng tương tự như vậy:



Hình 4. Đầu vào của thuật toán

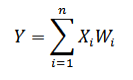
Perceptron nhận nhiều giá trị input binary và chỉ cho ra một kết quả đầu ra (output) duy nhất.

* + - Mỗi input tương ứng là một thuộc tính của dữ liệu.
    - Đầu ra (output) Là kết quả của mạng Perceptron, đầu ra tương ứng khi nhận vào các đầu vào inputs.
    - Các trọng số (weights) ứng với từng đầu vào.

Đây là thành phần quan trọng của mạng Perceptron nói chung và mạng neural nhân tạo nói riêng. Nó thể hiện mức độ quan trọng của từng input trong việc hình thành output. Quá trình học của mạng Perceptron cũng chính là quá trình thay đổi các trọng số sao cho thu được đầu ra gần với mong đợi nhất.

* + - Bộ cộng

Tính tổng trọng số của tất cả các input được đưa vào. Hàm tổng được tính như sau:



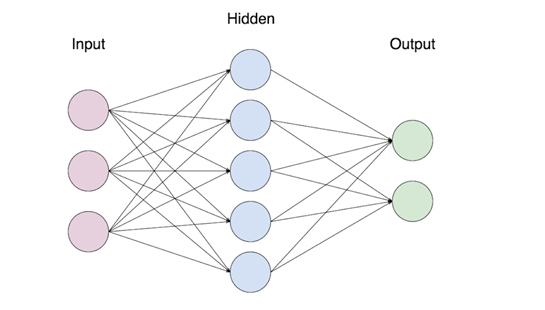
* + - Hàm chuyển đổi (Transfer function)

Hàm tổng (Summation Function) của một Neuron cho biết khả năng kích hoạt (Activation) của neuron đó còn gọi là kích hoạt bên trong (internal activation). Mối quan hệ giữa Internal Activation và kết quả (output) được thể hiện bằng hàm chuyển đổi (Transfer Function)

Việc lựa chọn Transfer Function có tác động lớn đến kết quả của ANN. Hàm chuyển đổi phi tuyến được sử dụng phổ biến trong ANN là sigmoid (logical activation) function.



*Mô hình tổng quát của Neural Network:*



Hình 5. Mô hình tổng quát của Neural Network

Hình trên là biểu diễn cơ bản của Neural Network, bao gồm Input layer, các hidden layer ở giữa và một output layer. Số layer được tính bằng công thức (số layer – 1) (không tính Input layer). Mỗi hidden layer bao gồm các đơn vị Node(biểu diễn bằng các vòng tròn trong hình). Mỗi node trong một layer thực hiện các bước:

* + - Liên kết với toàn bộ các node của layer trước đó
    - Mỗi node có hệ số bias b riêng
    - Tính tổng linear và sử dụng activation function.
* **Thuật toán *K-nearest neighbors***

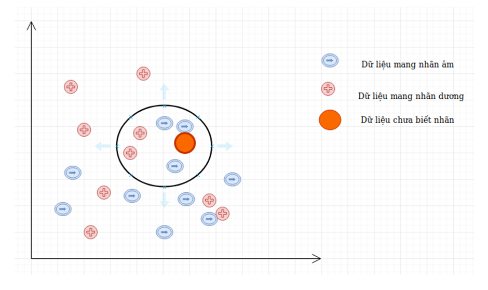
*Khái niệm K-nearest neighbors:*

Thuật toán KNN là một trong những phương pháp học có giám sát “Supervised Learning” tức dựa trên biến mục tiêu đã được xác định trước đó, thuật toán sẽ xem xét dữ liệu đã chứa biến mục tiêu (đã phân loại) để “học” và tìm ra những biến d có thể tác động đến biến mục tiêu.

KNN dựa trên giả định là những thứ tương tự hay có tính chất gần giống nhau sẽ nằm ở vị trí gần nhau, với giả định như vậy, KNN được xây dựng trên các công thức toán học phục vụ để tính khoảng cách giữa 2 điểm dữ liệu (gọi là Data points) để xem xét mức độ giống nhau của chúng.

KNN còn gọi là “Lazy learning method” vì tính đơn giản của nó, có nghĩa là quá trình training không quá phức tạp để hoàn thiện mô hình (tất cả các dữ liệu đào tạo có thể được sử dụng để kiếm tra mô hình KNN). Điều này làm cho việc xây dựng mô hình nhanh hơn nhưng giai đoạn thử nghiệm chậm hơn và tốn kém hơn về một thời gian và bộ nhớ lưu trữ, đặc biệt khi bộ dữ liệu lớn và phức tạp với nhiều biến khác nhau. Trong trường hợp xấu nhất, KNN cần thêm thời gian để quét tất cả các điểm dữ liệu và việc này sẽ cần nhiều không gian bộ nhớ hơn để lưu trữ dữ liệu. Ngoài ra KNN không cần dựa trên các tham số khác nhau để tiến hành phân loại dữ liệu, không đưa ra bất kỳ kết luận cụ thể nào giữa biến đầu vào và biến mục tiêu, mà chỉ dựa trên khoảng cách giữa data point cần phân loại với data point đã phân loại trước đó. Đây là một đặc điểm cực kỳ hữu ích vì hầu hết dữ liệu trong thế giới thực tại không thực sự tuân theo bất kỳ giả định lý thuyết nào ví dụ như phân phối chuẩn trong thống kê.

*Mô hình minh họa*:



Hình 6. Mô hình thuật toán K - Nearest Neighbors

Ví dụ:

Giả sử ta có D là tập các dữ liệu đã được phân loại thành 2 nhãn (+) và (-) được biểu diễn trên trục tọa độ như hình vẽ và một điểm dữ liệu mới A chưa biết nhãn. Vậy làm cách nào để chúng ta có thể xác định được nhãn của A là (+) hay (-)?

Có thể thấy cách đơn giản nhất là so sánh tất cả các đặc điểm của dữ liệu A với tất cả tập dữ liệu học đã được gắn nhãn và xem nó giống cái nào nhất, nếu dữ liệu (đặc điểm) của A giống với dữ liệu của điểm mang nhãn (+) thì điểm A mang nhãn (+), nếu dữ liệu A giống với dữ liệu nhãn (-) hơn thì nó mang nhãn (-), trông có vẻ rất đơn giản nhưng đó là những gì mà KNN làm.

Trong trường hợp của KNN, thực tế nó không so sánh dữ liệu mới (không được phân lớp) với tất cả các dữ liệu khác, thực tế nó thực hiện một phép tính toán học để đo khoảng cách giữa dữ liệu mới với tất cả các điểm trong tập dữ liệu học D để thực hiện phân lớp. Phép tính khoảng cách giữa 2 điểm có thể là Euclidian, Manhattan, trọng số, Minkowski, …

* **Thuật toán Ensemble Voting**:

*Khái niệm:*

Voting ensemble (còn gọi là “majority voting ensemble”) là một mô hình học máy tập hợp kết hợp các dự đoán từ nhiều mô hình khác.

Đây là một kỹ thuật có thể được sử dụng để cải thiện hiệu suất mô hình, lý tưởng đạt được hiệu suất tốt hơn bất kỳ mô hình đơn lẻ nào được sử dụng trong tập hợp.

Voting ensemble hoạt động bằng cách kết hợp các dự đoán từ nhiều mô hình. Nó có thể được sử dụng để phân loại hoặc hồi quy. Trong trường hợp hồi quy (regression), điều này liên quan đến việc tính trung bình của các dự đoán từ các mô hình. Trong trường hợp phân loại (classification), các dự đoán cho mỗi nhãn được tổng hợp và nhãn có đa số phiếu được dự đoán.

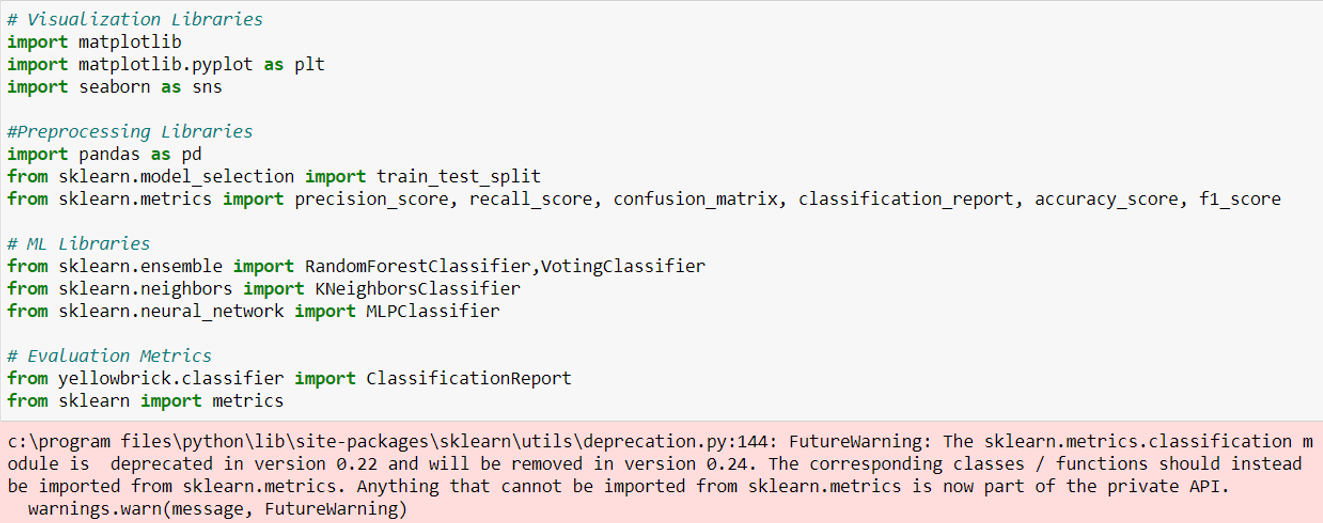
* Regression Voting Ensemble: Dự đoán dựa trên mức trung bình của các mô hình đóng góp.
* Classification Voting Ensemble: Dự đoán dựa trên phần lớn phiếu bầu của các mô hình đóng góp.

Có hai cách tiếp cận để dự đoán dựa trên phiếu bầu đa số để phân loại (Classification Voting Ensemble); đó là bỏ phiếu cứng (**Hard Voting**) và bỏ phiếu mềm (**Soft Voting**).

* Hard Voting: Dự đoán lớp dựa trên tổng số phiếu bầu lớn nhất từ ​​các mô hình.
* Soft Voting: Dự đoán lớp dựa trên xác suất tổng hợp lớn nhất từ ​​các mô hình.

### Tiền xử lý dữ liệu:

Cài đặt đặt và nhập các thư viện trực quan (matplotlib, seaborn), thư viện tiền xử lý dữ liệu và số (pandas, sklearn), thư viện yellowbrick.



Đọc dữ liệu đầu vào và kết nối dữ liệu:

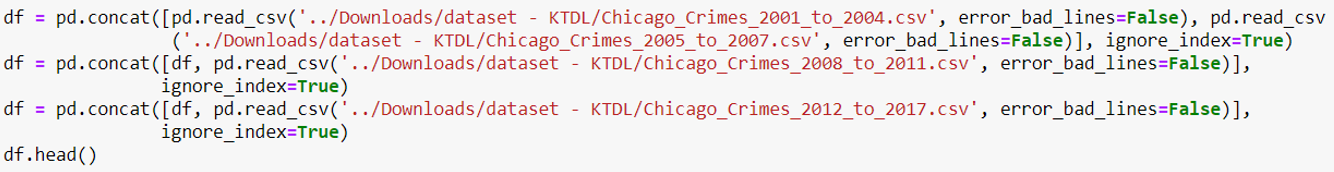
Vì bộ dữ liệu gồm 4 file “csv” nên ta cần đọc dữ liệu và kết nối chúng thành một thể thống nhất để có một Dataframe như mong muốn, phục vụ cho việc khai thác dữ liệu.

* Phương pháp: ​​​​​​pd.concat*( objs, axis = 0, join='outer', join\_axes = None, ignore\_index = False, keys = None, levels = None, names = None, verify\_integrity = False, copy = True)*

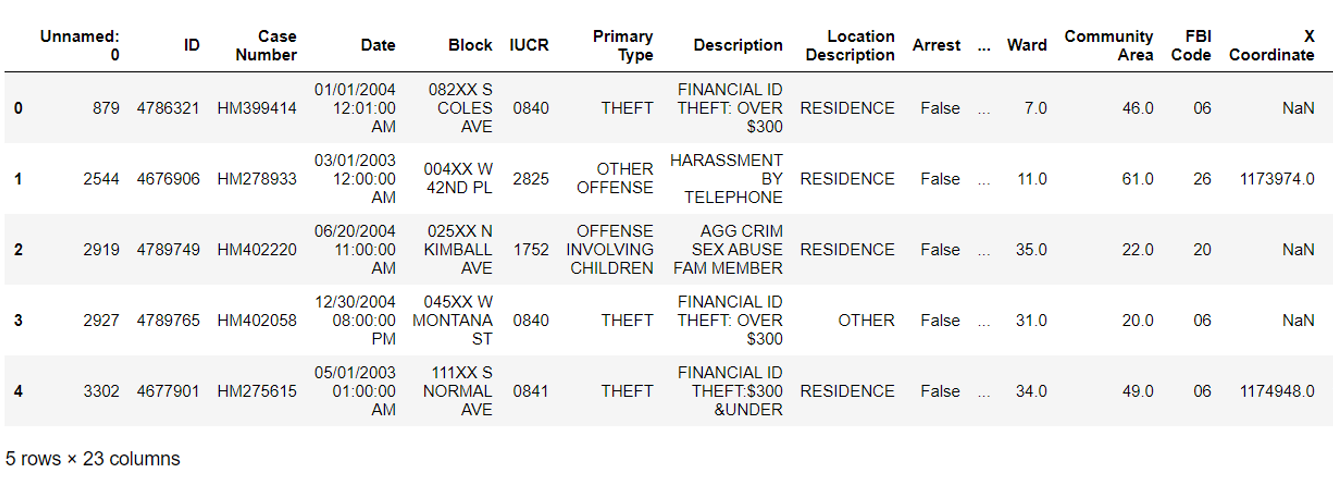
Lưu ý: sử dụng từ khóa **ignore\_index = True** để đảm bảo sau khi ghép nối, các nhãn chỉ mục của bảng mới được reset để loại bỏ nhãn chỉ mục bị lặp.

pd.concat(). Linh hoạt và có thể làm tốt cả việc ghép nối theo hàng và cột.

* Thao tác:



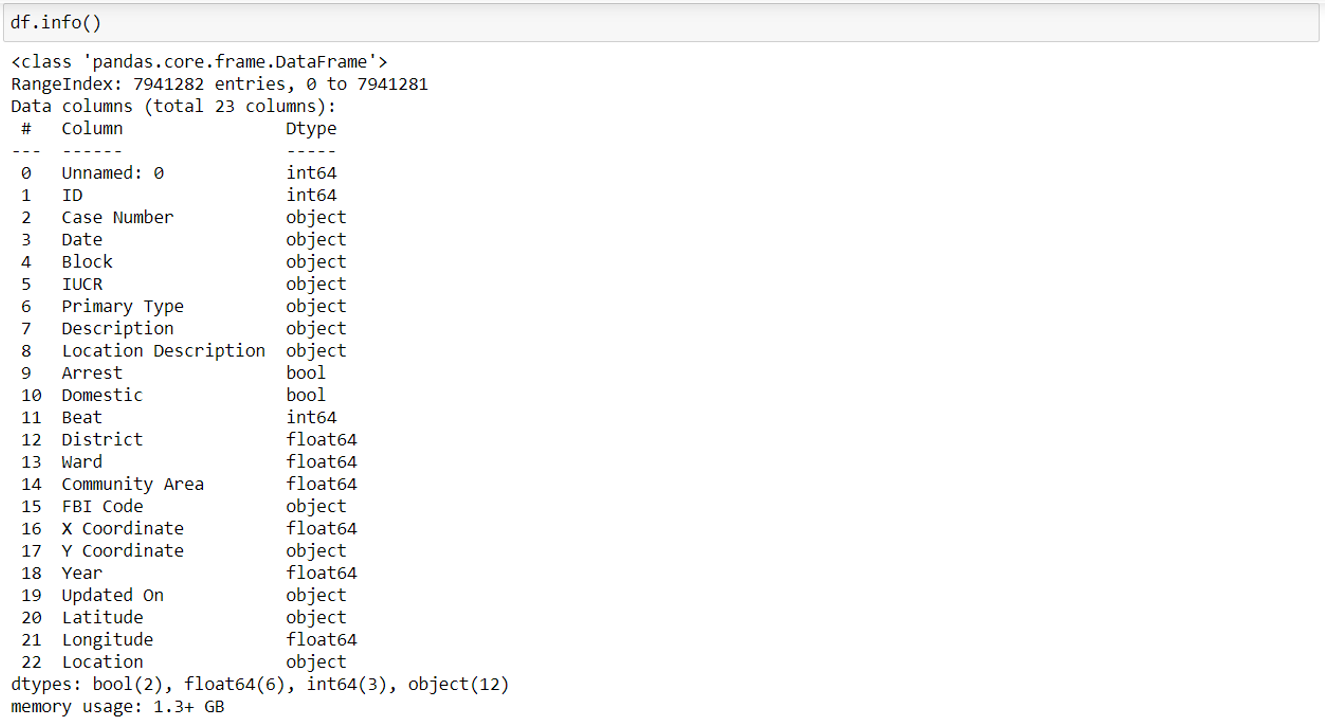
* Kết quả:



* **Xem xét dữ liệu đầu vào**:

Việc xem xét dữ liệu đầu vào giúp ta có cái nhìn tổng quát về bộ dữ liệu: số cột, số dòng, loại của từng thuộc tính, dung lượng của bộ dữ liệu.

* Phương pháp: DataFrame.info()
* Thao tác và kết quả:



* **Xóa các dữ liệu rỗng**:

Sau khi xem xét, ta thấy được bộ dữ liệu rất lớn, và tồn tại nhiều giá trị rỗng. Các giá trị rỗng này có thể gây ra một số vấn đề. Vấn đề rõ ràng nhất là không có đủ dữ liệu để chạy phân tích. Chính vì vậy ta cần xóa các giá trị rỗng, nhằm tăng độ chính xác cho bộ dữ liệu.

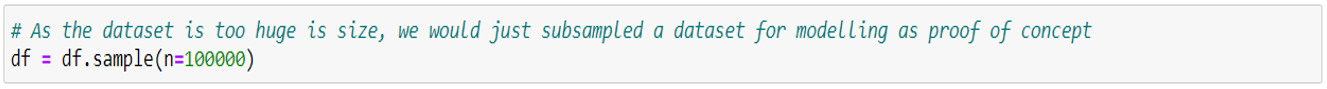
* Phương pháp: DataFrame.dropna( self, axis = 0, how = 'any', thresh = None, subset = None, inplace = False)
* Thao tác:



* **Thu giảm bộ dữ liệu**:

Bộ dữ liệu đầu vào gồm 23 cột và gần 8 triệu dòng, chính vì bộ dữ liệu quá lớn nên việc khai thác trở nên vô cùng khó khăn và mất nhiều thời gian. Chính vì vậy ta cần thu giảm dữ liệu gốc bằng các giảm bớt các dòng dữ liệu ( còn 100000 dòng). Tập dữ liệu được biến đổi đảm bảo các toàn vẹn, nhưng nhỏ/ít hơn nhiều về số lượng so với ban đầu.

* Phương pháp: DataFrame.sample( *self: ~FrameOrSeries, n = None, frac = None, replace = False, weights = None, random\_state = None, axis = None*)
* Thao tác:



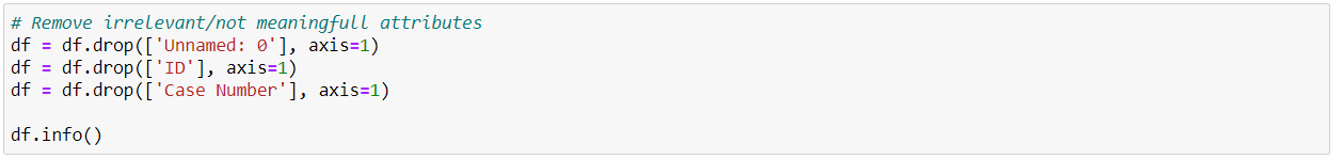
* **Xóa các thuộc tính không liên quan hoặc không có ý nghĩa**.

Đối với các thuộc tính không liên quan hoặc không có ý nghĩa trong mục đích khai thác, ta nên xóa thuộc tính đó đi, điều này làm cho bộ dữ liệu được tối ưu hơn, tiết kiệm được thời gian khai thác.

* Phương pháp: DataFrame.drop(self, labels = None, axis = 0, index = None, columns = None, level = None, inplace = False, errors = 'raise')

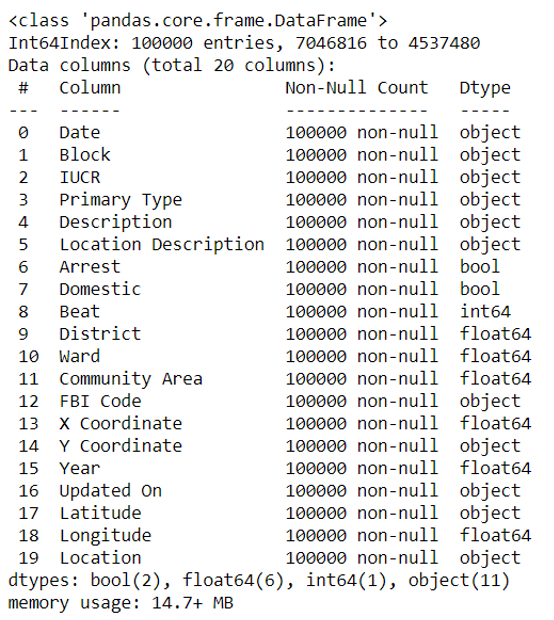
Lưu ý: sử dụng từ khóa axis = 1 là để xóa cột, còn axis = 0 là để xóa chỉ mục(index).

* Thao tác:



Trong bộ dữ liệu trên, ta sử dụng hàm dataframe.drop() để xóa các thuộc tính không cần thiết là thuộc tính: Unnamed, ID, Case Number.

* Kết quả:



* **Tách thuộc tính “Date” thành các thuộc tính “ Year”, “Month”, “Day”, “Hour”, “Minute”, “Second”.**

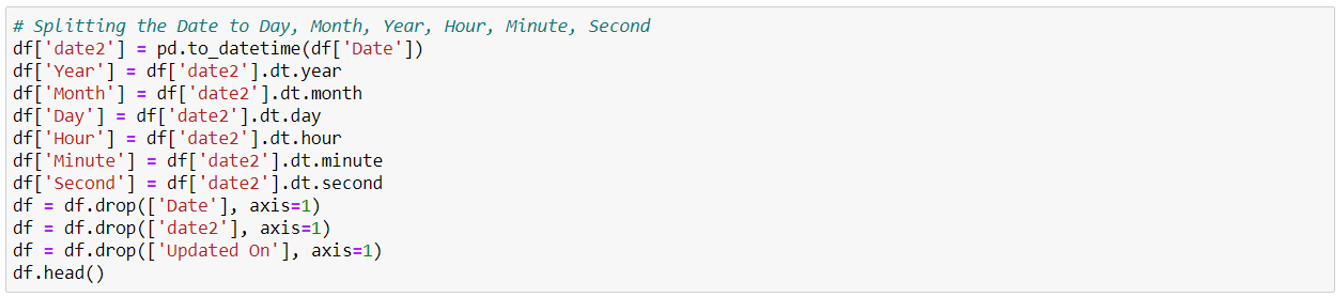
Trong bộ dữ liệu mẫu, thuôc tính “Date” gồm nhiều giá trị như ngày, tháng, năm, giờ, phút, giây, chính vì vậy ta cần phải tách các giá trị này thành các thuộc tính độc lập nhằm khai thác dữ liệu, truy xuất các thuộc tính cũng như giá trị thuộc tính dễ dàng hơn.

Sau khi tách thuộc tính “Date” thành các thuộc tính “ Year”, “Month”, “Day”, “Hour”, “Minute”, “Second” thì ta nên xóa thuộc tính “Date” để tránh việc trùng lặp.

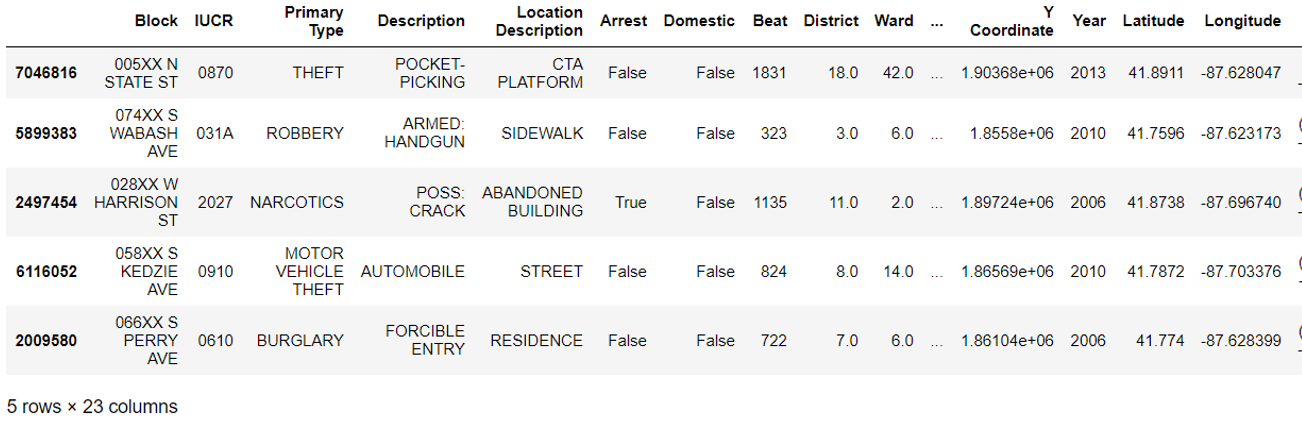
* Phương pháp: Chuyển đổi đối số thành Datetime:

pandas.to\_datetime *(arg, errors = ‘raise’, dayfirst = False, yearfirst = False, utc = None, format = None, exact = True, unit = None, infer\_datetime\_format = False, origin = ‘unix’, cache = True*)

* Thao tác:



* Kết quả:



* **Chuyển đổi các thuộc tính phân loại thành số**:

Mã hóa đối tượng dưới dạng kiểu liệt kê hoặc biến phân loại.

Phương pháp này hữu ích để có được một biểu diễn số của một mảng khi tất cả những gì quan trọng là xác định các giá trị riêng biệt (giá trị phân loại). Factize có sẵn dưới dạng cả hàm cấp cao pandas.factorize () và như một phương thức Series.factorize () và Index.factorize ().

* Phương pháp:

pandas.factorize(*values,sort: bool = False, na\_sentinel: int = -1, size\_hint:Union[int, NoneType] = None* )

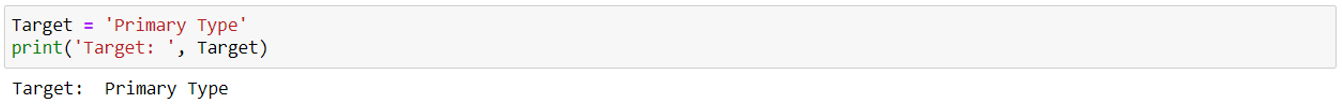
* Thao tác:



* **Xác định Thuộc tính quyết định (đầu ra) của dữ liệu**:

Mục tiêu của bài toán là dự đoán phân loại tội phạm, nên ta chọn thuộc tính “Primary Type” làm thuộc tính quyết định.

* Thao tác:



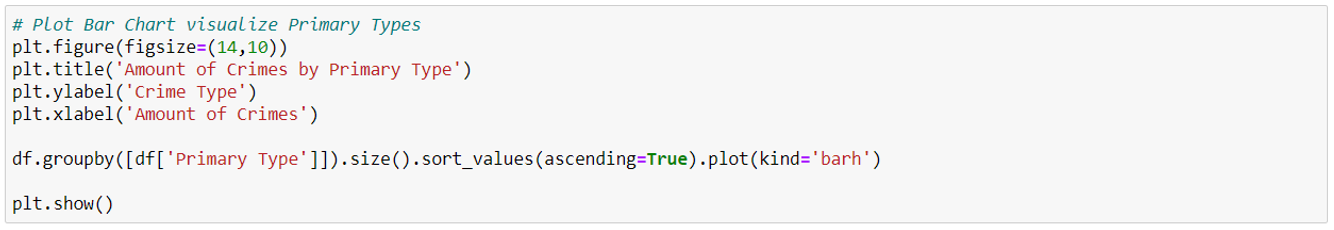
* **Vẽ biểu đồ thanh (Plot Bar Chart)**

Sau khi xác định thuộc tính quyết định là “Primary Type”, ta cần có cái hình khái quát và tổng thể hơn về các giá trị trong thuộc tính này, bằng cách vẽ biểu đồ thanh, biểu đồ này sẽ cho ta thấy các phân loại trong Loại tội phạm.

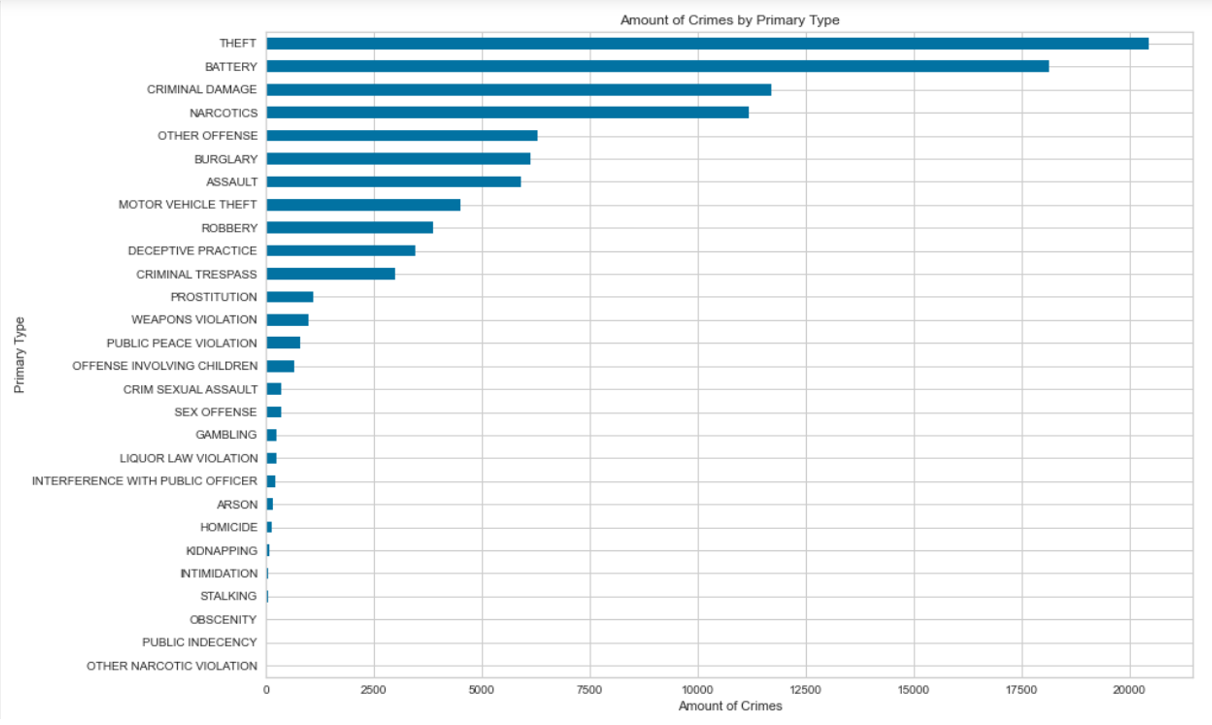
Biểu đồ thanh là biểu đồ thể hiện dữ liệu phân loại với các thanh hình chữ nhật có chiều cao hoặc chiều dài tỷ lệ với các giá trị mà chúng đại diện. Các thanh có thể được vẽ theo chiều dọc hoặc chiều ngang.

Một biểu đồ thanh hiển thị so sánh giữa các loại riêng biệt. Một trục của biểu đồ hiển thị các danh mục cụ thể được so sánh và trục còn lại biểu thị giá trị đo.

* Thao tác:



* Kết quả:



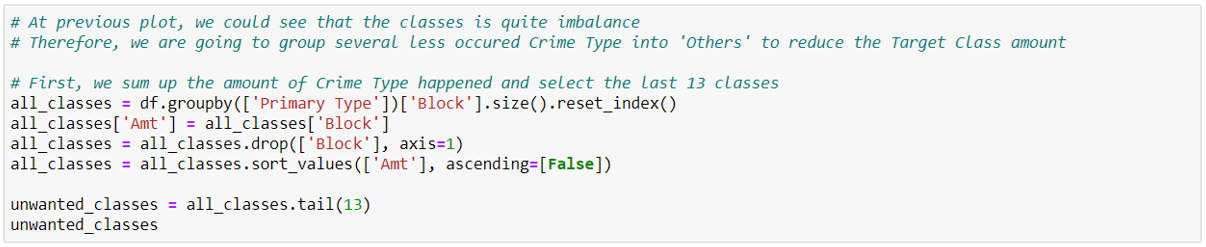
Qua biểu đồ thanh, ta có thể thấy 28 giá trị phân loại trong thuộc tính Primary Type, trong đó giá trị “Theft” và “Battery” chiếm tỷ lệ phân loại cao nhất: với “Theft” chiếm hơn 20000 dòng, “Battery” hơn 17500 dòng dữ liệu.

* **Gom các Loại tội phạm ít xảy ra**:

Ở biển đồ thanh trước, chúng ta có thể thấy rằng các lớp khá mất cân đối, bởi vì phân loại tội phạm không đồng đều, chủ yếu tập trung ở các loại như “Theft”, “Battery”, “Criminal damage”, “Narcotics”,… trong khi đó các loại như “Obscenity”, “Public indecency”, “Other narcotic violation”… lại rất ít. Chính vì vậy ta cần gom các Loại tội phạm ít xảy ra vào một Loại là “Others” để giảm số lượng lớp mục tiêu.

Đầu tiên, chúng ta tổng hợp số lượng Loại tội phạm ít xảy ra để tiến thành bước tiếp theo là gom chúng lại. Ta chọn 13 lớp cuối cùng trong bảng đồ thanh.

* Thao tác:

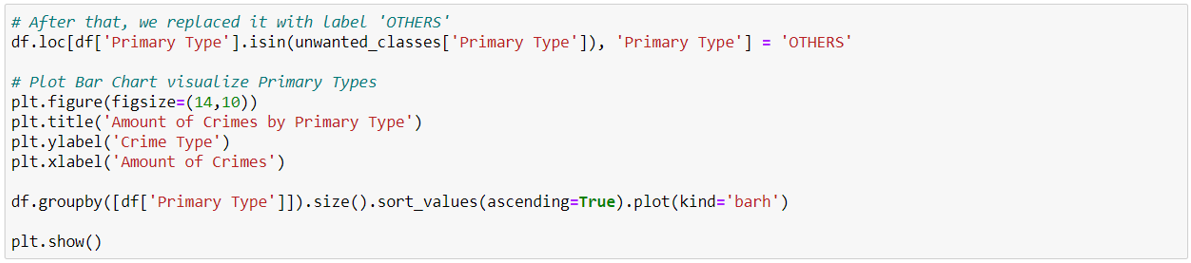


* Kết quả sau khi gom:

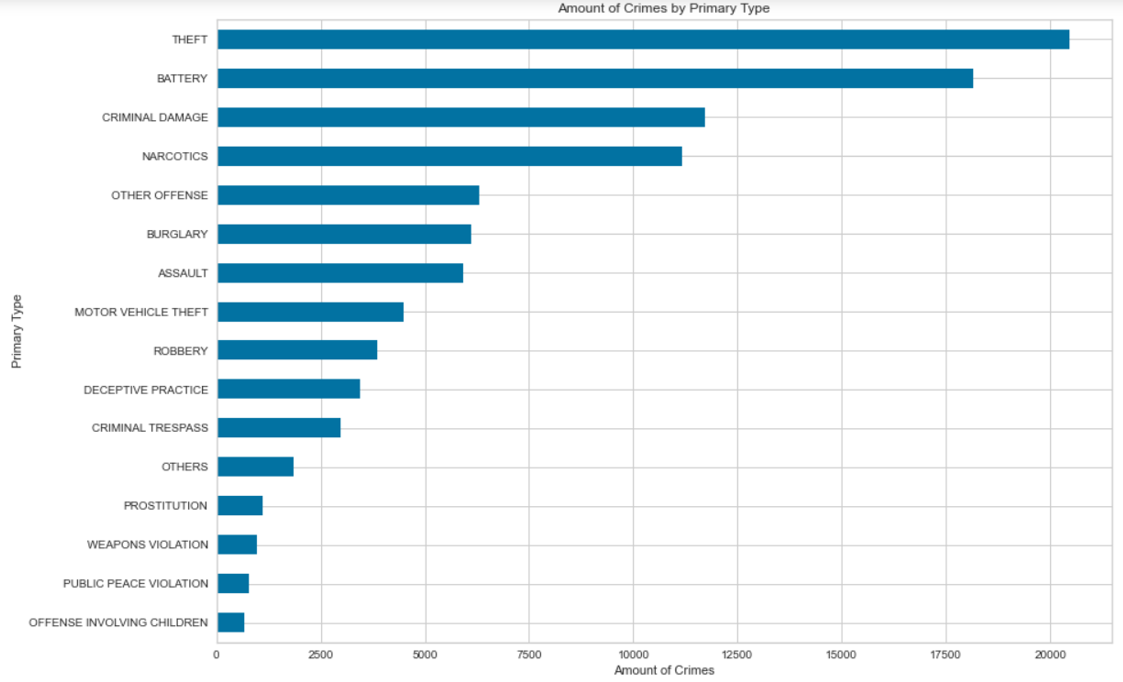
A screenshot of a cell phone

Description automatically generated

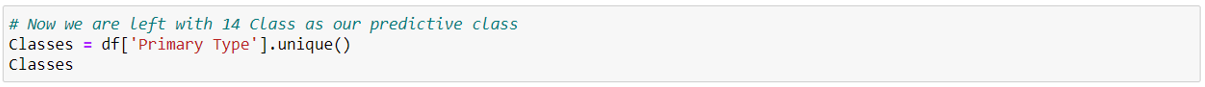
* **Tiến hành thay thế bằng một nhãn “Others” đối với 13 lớp trên**.
* Thao tác:



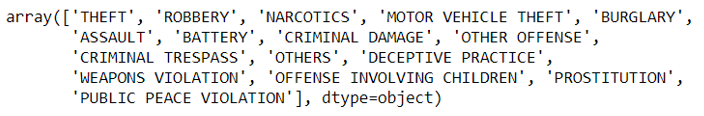
* Kết quả:



* Bây giờ, chúng ta còn lại 16 lớp phân lớp trong Loại tội phạm.
* Phương pháp : dataframe.unique()
* Thao tác:



* Kết quả:



* **Chuyển đổi 16 lớp phân loại tội phạm thành số**:

Mã hóa đối tượng dưới dạng kiểu liệt kê hoặc biến phân loại.

Phương pháp này hữu ích để có được một biểu diễn số của một mảng khi tất cả những gì quan trọng là xác định các giá trị riêng biệt (giá trị phân loại).

* Phướng pháp: pandas.factorize(*values,sort: bool = False, na\_sentinel: int = -1, size\_hint:Union[int, NoneType] = None* )

Dataframe.unique()

* Thao tác:



* Kết quả:



* **Lựa chọn tính năng bằng phương thức lọc/ Chia DataFrame để xác định mục tiêu lớp và tính năng.**
* Thao tác:



* **Sử dụng hệ số tương quan Pearson**

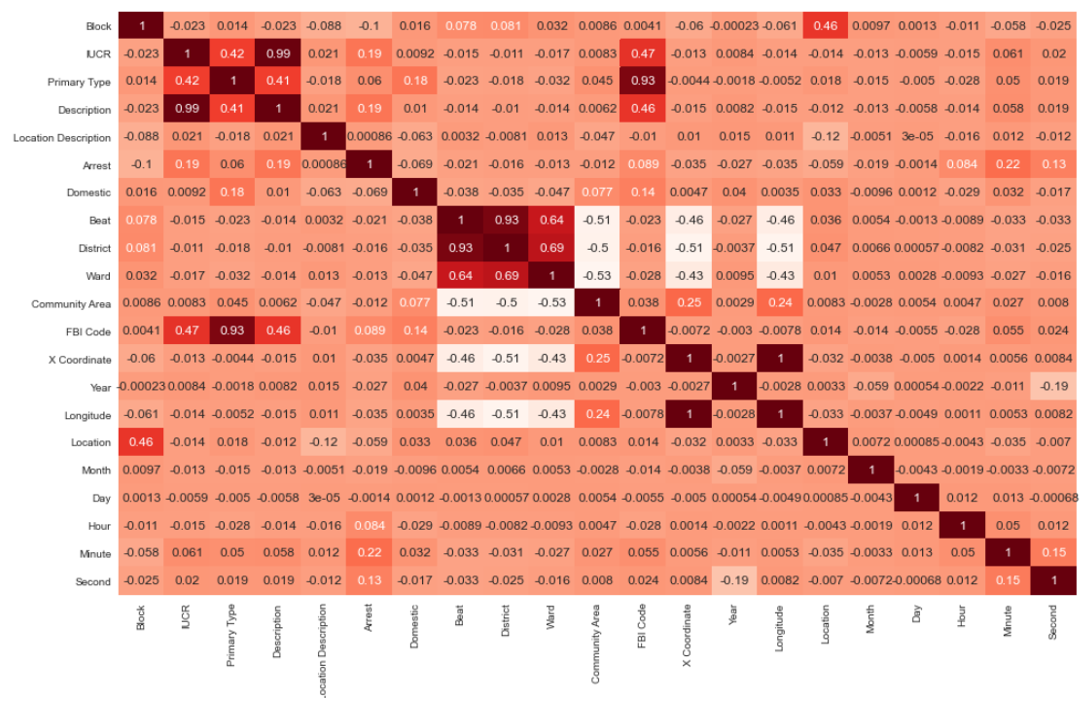
***Hệ số tương quan Pearson*** (Pearson correlation coefficient, ký hiệu r) là số liệu thống kê kiểm tra đo lường mối quan hệ thống kê hoặc liên kết giữa các biến phụ thuộc với các biến liên tục.

Tương quan pearson được biết đến như là phương pháp tốt nhất để đo lường mối liên hệ giữa các biến quan tâm bởi vì nó dựa trên phương pháp hiệp phương sai. Nó cung cấp thông tin về mức độ quan trọng của mối liên hệ, hoặc mối tương quan, cũng như hướng của mối quan hệ. Ngoài ra, việc kiểm tra ***hệ số tương quan pearson*** còn giúp chúng ta sớm nhận diễn được sự xảy ra của vấn đề đa cộng tuyến khi các biến độc lập có sự tương quan mạnh với nhau.

* Thao tác:



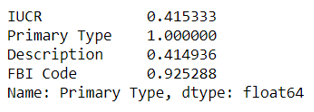
* Kết quả:



* Hệ số tương quan có giá trị từ -1 đến 1.
* Hệ số tương quan bằng 0 (hay gần 0) có nghĩa là hai biến số không có liên hệ gì với nhau.
* Hệ số bằng -1 hay 1 có nghĩa là hai biến số có một mối liên hệ tuyệt đối.
* Hệ số tương quan là âm (r <0) có nghĩa là khi x tăng cao thì y giảm (và ngược lại, khi x giảm thì y tăng)
* Hệ số tương quan là dương (r > 0) có nghĩa là khi x tăng cao thì y cũng tăng, và khi x tăng cao thì y cũng tăng theo.
* **Lọc các thuộc tính có độ tương quan cao với thuộc tính quyết định (r>0.2)**
* Thao tác:



* Kết quả: Có 4 thuộc tính có độ tương quan cao với đầu ra là “IUCR”, “Primary Type”, “Description”, “FBI Code”.



* **Các thuộc tính được chọn thủ công dựa trên Phần lọc thuộc tính ở bước trên**
* Thao tác:



* **Chia bộ dữ liệu thành Training Set (Tập huấn luyện) và Test Set ( Tập kiểm chứng)**

Để đánh giá một mô hình phân tích, khai phá dữ liệu, thì việc đầu tiên, và thông thường nhất là chúng ta cần chia tập dữ liệu thành các phần phục vụ cho việc huấn luyện mô hình (Training Data ) và kiểm chứng mô hình (Test Data). Xây dựng model thông qua việc lựa chọn thuật toán, xác định các biến dữ liệu, hay điều chỉnh các tham số, v..v sao cho phù hợp sẽ dựa trên dữ liệu training. Sau khi hoàn thành cơ bản mô hình, chúng ta sử dụng dữ liệu test hay còn gọi là “unseen data” để thử nghiệm mô hình bằng cách kết hợp nhiều công thức, chỉ số (metrics) khác nhau để đánh giá mức độ chính xác với input là kết quả (kết quả dư báo hay phân loại) có được khi chạy mô hình với dữ liệu test.

* Thao tác: Chia bộ dữ liệu thành 20% Test và 80% Training.



# THUẬT TOÁN:

Như đã được đề cập ở trên thì các thuật toán khai thác dữ liệu được lựa chọn đó là Random Forest (RF) – Rừng ngẫu nhiên; Neural Network (NN) – Mạng nơ-ron nhân tạo; K - Nearest Neighbors (KNN) – k láng giềng gần nhất và cuối cùng là một thuật toán Ensemble, là sự kết hợp của 3 mô hình độc lập ở trên để tạo nên một mô hình mới kết hợp dự đoán của 3 mô hình trên để đưa ra dự đoán tổng thể cuối cùng.

### Lý do lựa chọn:

Để có thể hiểu rõ hơn về lý do các thuật toán trên được chọn, chúng ta sẽ bắt đầu đi từ phân tích bài toán. Bài toán đặt ra cho chúng ta đó là chúng ta cần phải dự đoán nhãn của các vụ án, mà cụ thể ở đây là loại tội phạm, của một dự liệu mới dựa trên dữ liệu đã được huấn luyện mà trong đó mỗi mẫu dữ liệu đều đã được gán nhãn. Có thể thấy rằng với bộ dữ liệu huấn luyện, tất cả các dòng dữ liệu đều được gán một nhãn liên quan đến loại tội phạm. Khi đó, thông qua một quá trình huấn luyện, một mô hình sẽ được xây dựng để cho ra các dự đoán và khi các dự đoán bị sai thì mô hình này sẽ được tinh chỉnh lại. Việc huấn luyện sẽ tiếp tục cho đến khi mô hình đạt được mức độ chính xác mong muốn trên dữ liệu huấn luyện. Chính vì đặc tính trên của bộ dữ liệu nên chúng ta phải sử dụng những thuật toán học có giám sát (Supervised Learning Algorithms) để có thể giải quyết bài toán.

Các thuật toán học có giám sát được phân ra thành hai loại chính là phân lớp (Classification) và hồi quy (Regression). Một bài toán được gọi là phân lớp nếu các nhãn của dữ liệu đầu vào được chia thành một số hữu hạn lớp (miền giá trị là rời rạc). Trong khi đó, một bài toán được xem là hồi quy nếu nhãn không được chia thành các nhóm mà là một giá trị thực cụ thể (miền giá trị là liên tục). Như có thể thấy, bài toán của chúng ta sẽ được gọi là phân lớp do các nhãn tội phạm là các miền giá trị rời rạc (cụ thể là 17 loại), chứ không liên tục. k-Nearest Neighbors, Neural Network và Random Forest đều là những thuật toán nổi tiếng dùng để giải quyết bài toán phân loại cho nên cả 3 được chọn.

Đầu tiên, với k-Nearest Neighbors, thì có thể nói rằng đây là mô hình thuật toán vô cùng đơn giản và không tốn quá nhiều thời gian để cài đặt. Trong một số trường hợp nhất định, việc áp dụng KNN để phân lớp có thể đem lại kết quả chính xác nhưng lại không tốn quá nhiều công sức. Ưu điểm của thuật toán này đó là độ phức tạp tính toán của quá trình training là bằng 0. Việc dự đoán kết quả của dữ liệu mới rất đơn giản. Không cần giả sử gì về phân phối của các class. Tuy nhiên KNN cũng có các nhược điểm sau đây: KNN rất nhạy cảm với nhiễu khi K nhỏ. Và KNN là một thuật toán mà mọi tính toán đều nằm ở khâu test. Trong đó việc tính khoảng cách tới từng điểm dữ liệu trong training set sẽ tốn rất nhiều thời gian, đặc biệt là với các cơ sở dữ liệu có số chiều lớn và có nhiều điểm dữ liệu. Với K càng lớn thì độ phức tạp cũng sẽ tăng lên. Ngoài ra, việc lưu toàn bộ dữ liệu trong bộ nhớ cũng ảnh hưởng tới hiệu năng của KNN. Chính vì vậy việc chỉ sử dụng KNN để phân lớp dữ liệu sẽ không đảm bảo được độ chính xác cao. Cho nên phải sử dụng kết hợp kết quả phân lớp của KNN với các mô hình thuật toán khác.

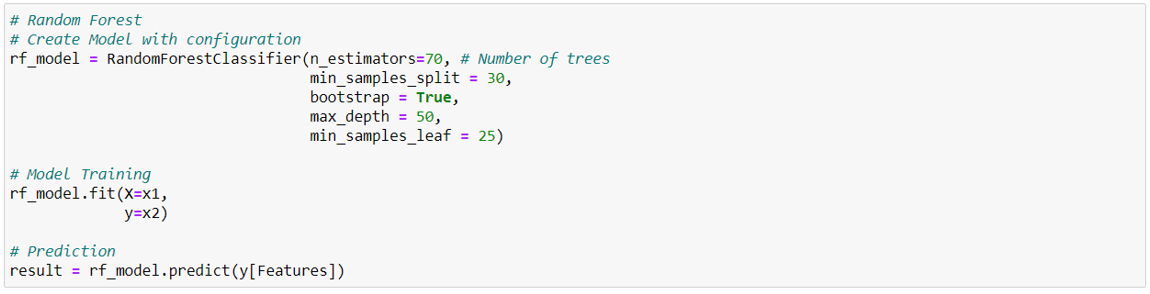
Tiếp theo, chúng ta sẽ đi tìm hiểu lý do lựa chọn thuật toán Neural Network. Như đã biết thì thuật toán Neural Network là một trong những thuật toán phổ biến nhất dùng để phân loại dữ liệu. Cách tổ chức của thuật toán này vô cùng phức tạp nhưng chặt chẽ nên dẫn đến là kết quả phân lớp của thuật toán này có độ chính xác rất cao. Neural Network là một trong những thuật toán Machine Learning mạnh nhất và nó phù hợp để giải quyết các bài toán có số lượng dữ liệu nhiều và số biến lớn. Rõ ràng so với KNN thì Neural Network sẽ đưa ra kết quả phân lớp tốt hơn nhiều. Tuy nhiên, Neural Network vẫn tồn tại nhiều nhược điểm. Thứ nhất, NN chỉ có thể làm việc với các dữ liêu số. Thứ hai, NN không thể đưa ra được cơ chế giải thích, đây là nhược điểm chính yếu nhất của mạng neural. Nhưng nhược điểm này đã được khắc phục đáng kể thông qua các giải thuật rút trích từ mạng. Thứ ba, khả năng xử lý và độ chính xác trong xử lý dữ liệu của mạng neural phụ thuộc khá lớn vào số lượng hidden layer (lóp ẩn) của mạng. Việc lựa chọn đúng số lượng lớp hidden layer và số lượng nút trong từng hidden layer là một câu hỏi lớn cần phải trả lời. Nếu chọn số lượng hidden layer quá lớn, số lượng hệ số cần tối ưu cũng lớn lên và mô hình sẽ trở nên phức tạp. Sự phức tạp này ảnh hưởng tới hai khia cạnh. Thứ nhất, tốc độ tính toán sẽ bị chậm đi rất nhiều. Thứ hai, nếu mô hình quá phức tạp, nó có thể biểu diễn rất tốt training data, nhưng lại không biểu diễn tốt test data. Hiện tượng này gọi là Overfitting. Còn nếu số lượng hidden layer quá nhỏ, thì độ chính xác của thuật toán sẽ không được đảm bảo.

Kế tiếp, chúng ta sẽ tìm hiểu thuật toán còn lại trong số các thuật toán đã được đề cập ở trên, đó chính là Random Forest. Thuật toán K- Nearest Neighbors mà chúng ta tìm hiểu ở trên là một thuật toán được sử dụng rất rộng rãi. Tuy nhiên, nó cũng là một thuật toán rất cũ. Cho đến thời điểm này, các nhà nghiên cứu đã và đang phát triển những thuật toán phức tạp và mạnh mẽ hơn nhiều. Trong số đó có thuật toán Random Forest. Đây là một thuật toán mới xuất hiện tầm 10 năm trước, và được coi là một cuộc cách mạng trong Machine Learning. Random Forest chỉ phức tạp hơn một chút so với k-nearest neighbors, nhưng nó hiệu quả hơn nhiều nếu xét trên hiệu năng tính toán của máy tính. Bên cạnh đó, Random Forest còn cho kết quả chính xác hơn nhiều so với k-nearest neighbors. Một trong những lí do khiến Random Forest hiệu quả hơn so với K- Nearest Neighbors là bởi vì, với K- Nearest Neighbors, tất cả các thuộc tính đều có mức độ quan trọng như nhau. Mặt khác, Random Forest có khả năng tìm ra thuộc tính nào quan trọng hơn so với những thuộc tính khác. Trên thực tế, nó còn có thể chỉ ra rằng một số thuộc tính là hoàn toàn vô dụng. Bản chất Random Forest là một tập hợp mô hình (ensemble). Mô hình Random Forest rất hiệu quả cho các bài toán phân loại vì nó huy động cùng lúc hàng trăm mô hình nhỏ hơn bên trong với quy luật khác nhau để đưa ra quyết định cuối cùng. Mỗi mô hình con có thể mạnh yếu khác nhau, nhưng theo nguyên tắc “wisdom of the crowd”, ta sẽ có cơ hội phân loại chính xác hơn so với khi sử dụng bất kì một mô hình đơn lẻ nào. Random Forest đưa ra kết quả có độ chính xác rất cao, nhưng đổi lại ta không thể nào hiểu được cơ chế hoạt động bên trong mô hình vì cấu trúc quá phức tạp. RF do đó là một trong số những mô hình hộp đen (black box).

Cả 3 mô hình thuật toán kể trên đều có những điểm mạnh và điểm yếu khác nhau. Khi là những mô hình độc lập thì bản thân NN hay RF đã có độ chính xác khá cao rồi. Nhưng để tăng độ chính xác thêm một ít và tận dụng được các điểm mạnh của các mô hình độc lập, nhóm đề xuất một tập hợp mô hình (ensemble) là sự kết hợp của 3 mô hình độc lập kể trên. Trong mô hình này cũng sẽ sử dụng cơ chế bầu chọn theo số đông giống như RF, và trọng số của cả 3 mô hình độc lập là như nhau. Và như đã giải thích ở trên, mô hình kết hợp này sẽ giúp chúng ta tăng cơ hội phân loại chính xác hơn so với khi sử dụng bất kỳ một mô hình dơn lẻ nào.

### Các thuật toán:

#### Random Forest:



Để tạo mô hình Random Forest, chúng ta sẽ sử dụng hàm **RandomForest Classifier** của thư viện **sklearn** để khởi tạo mô hình. Các thông số của hàm khởi tạo như sau:

* + **N – estimator** (kiểu số nguyên int, mặc định là 100): Là số cây sẽ được tạo trong mô hình. Ở đây là 70 cây.
  + **Min\_samples\_split** (kiểu số nguyên int hoặc số thực float, mặc định là 2): Số lượng mẫu tối thiểu cần thiết để phân chia một nút nội bộ. Ở đây là 30 mẫu.
  + **Bootstrap** (kiểu bool, mặc định là True): Quyết định xem có sử dụng phương pháp bootstrap để tạo mẫu xây dựng cây hay không. Nếu giá trị là False, sẽ sử dụng cả dataset để xây dựng từng cây. Về phương pháp bootstrapping, thì nó được sử dụng để giải quyết một vấn đề khá lớn của thuật toán Random Forest. Đó chính là việc nếu như tất cả các cây được dựng theo cùng một cách, chúng sẽ cho những câu trả lời giống nhau. Như vậy chẳng khác gì chúng ta chỉ sử dụng một cây quyết định duy nhất cả. Phương pháp bootstrapping sẽ giải quyết điều này bằng cách chọn ngẫu nhiên các quan sát (observations). Chính xác hơn, nó sẽ xoá một số quan sát và lặp lại một số khác một cách ngẫu nhiên. Xét toàn cục, những quan sát này vẫn rất gần với tập các quan sát ban đầu, nhưng những thay đổi nhỏ sẽ đảm bảo rằng mỗi cây quyết định sẽ có một chút khác biệt.
  + **Max\_depth** (kiểu int, mặc định là không có): Độ dài tối đa của cây. Nếu không đặt giá trị này, thì các nút được mở rộng cho đến khi tất cả các nút trở thành nút lá hoặc cho đến khi tất cả các lá chứa ít hơn các mẫu min\_samples\_split. Ở đây đặt giá trị là 50.
  + **Min\_samples\_leaf** (kiểu int hoặc float, mặc định là 1): Số lượng mẫu tối thiểu cần có tại một nút lá. Một điểm phân chia ở bất kỳ độ sâu nào sẽ chỉ được xem xét nếu nó để lại số mẫu đào tạo tối thiểu min\_samples\_leaf ở mỗi nhánh trái và phải. Điều này có thể có tác dụng làm mịn mô hình, đặc biệt là trong hồi quy. Ở đây giá trị là 25.

Sau đó, chúng ta sẽ tiến hành đào tạo mô hình trên tập huấn luyện và thực hiện các dự đoán trên tập kiểm tra.

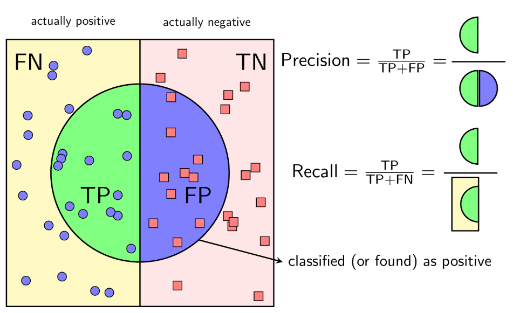


Ở đây, hàm **Accuracy\_score** cho biết độ chính xác của biến dự đoán đối với số liệu thực tế. Hàm **Precision\_score** cho biết tỉ lệ số điểm **true positive** trong số những điểm được phân loại là **positive** (TP + FP) và hàm **Recall\_score** được định nghĩa là tỉ lệ số điểm **true positive** trong số những điểm **thực sự là *positive*** (TP + FN).

Giá trị mang nhãn positive (True positive) tức các dữ liệu đúng và được phân lớp đúng vào lớp đúng, các giá trị mang nhãn negative (True negative) là các dữ liệu sai và được phân lớp đúng vào lớp sai. Ngoài ra, giá trị mang nhãn False Positive là các giá trị đúng bị phân lớp sai vào lớp sai và giá trị mang nhãn False Negative là các giá trị sai bị phân lớp sai vào lớp đúng.

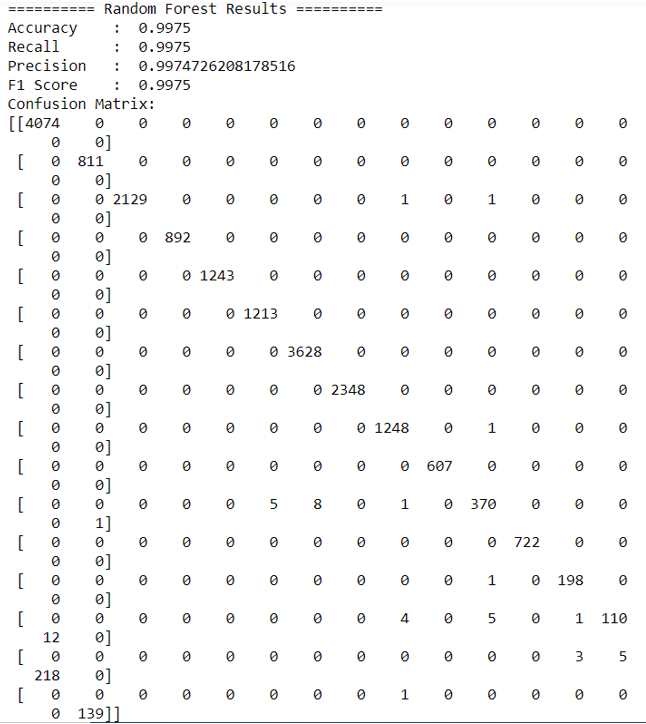
Chúng ta còn có hàm F1\_score – giá trị tiên quyết của thuật toán dự báo. Giá trị này cho biết giá trị bình quân điều hòa của Precision và Recall. Nói chung, đại lượng này là đại lượng quyết định tính chính xác của mô hình thuật toán khai thác dữ liệu được đặt ra.

Tất cả các độ đo này được tính toán để xác định ma trận nhầm lẫn (Confussion Matrix) của bài toán.



Sau khi đào tạo, chúng ta sẽ tiến hành kiểm tra tính chính xác của mô hình bằng ma trận nhầm lẫn.

* Kết quả:



Kết quả hiển thị sau khi tính toán gồm các giá trị sau:

**Accuracy** = 0.9975 ~ 1 Điều này cho thấy phần trăm dự đoán đúng là xấp xỉ 100%. Tức sử dụng thuật toán này đưa ra dự đoán có độ chính xác rất cao.

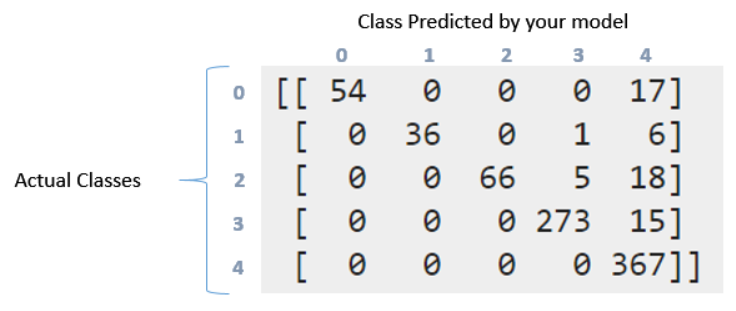
Tương tự như vậy, các giá trị **Precision, Recall** và **F1 score** đều có kết quả rất cao, lần lượt là 0.9975, 0.9975 và 0.9975.

Đối với giá trị Precision, khi giá trị kiểm dịnh được bằng 1 thì có nghĩa là mọi điểm thực hiện việc dự báo đều thực sự là positive, tức là không có điểm negative nào lẫn vào kết quả. Tuy nhiên, nếu chỉ Precision có giá trị bằng 1 thì kết quả kiểm định được không thực sự tốt do có thể chỉ kiểm được một điểm positive trong bài toán. Như vậy, giá trị Precision này lại không mang ý nghĩa bao quát cho cả tập dữ liệu.

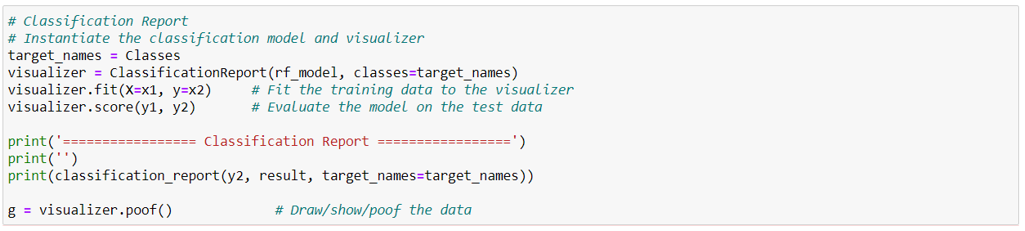
Đối với giá trị Recall, khi giá trị kiểm định được bằng 1 tức là mọi điểm positive đều đã được tìm thấy. Nhưng kết quả này có thể bao hàm nghĩa là có điểm negative bị lẫn trong đó. Tức đây là tập dữ diệu được dự đoán không tốt.

Tuy nhiên, khi cả hai giá trị trên cùng cao ( = 1 ) thì đây đúng là kết quả cho thấy tập dữ liệu này là tập dữ liệu tốt. Điều này có nghĩa là thuật toán này đưa ra tập dự liệu được dự báo chính xác và ít nhiễu.

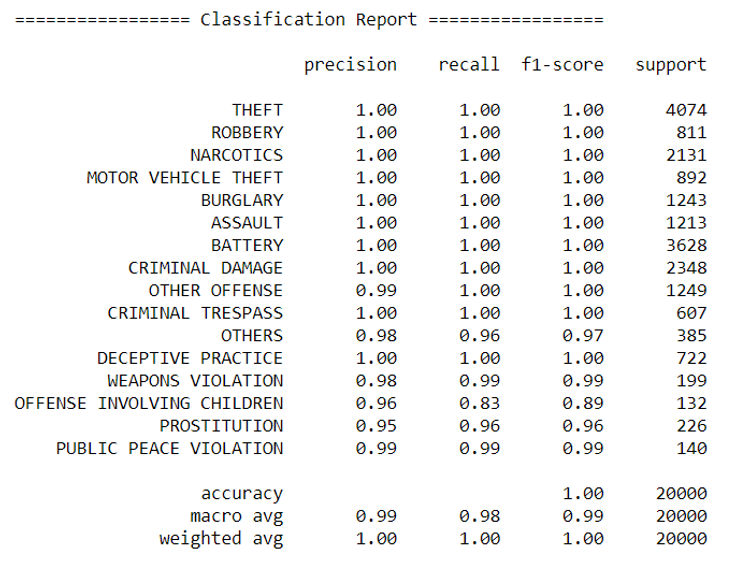
Quay trở lại kết quả của ma trận nhầm lẫn nhận được, ta thấy rõ ràng các kết quả trên đường chéo chính lớn hơn các giá trị ngoài đường chéo rất nhiều. Điều này cho thấy thuật toán được ta đưa vào kiểm định là thuật toán phù hợp với bộ dữ liệu này. Có rất ít dữ liệu bị phân lớp nhầm. Ví dụ như bộ dữ liệu thứ ba (nhãn 3) chỉ có 2 trong số các giá trị bị phân nhầm lớp. Ta có thể biết điều đó dựa vào quy cách sau:



Sau đó, ta tiến hành lập báo cáo phân loại đơn giản cho tập dữ liệu đã kiểm định ở trên để xác định rõ ràng hơn độ chính xác của từng giá trị thuộc tính của bộ dữ liệu.



* Kết quả



Ở đây, kết quả chạy báo cáo phân loại cho thấy độ chính xác của từng giá trị thuộc tính tội phạm được xác định.

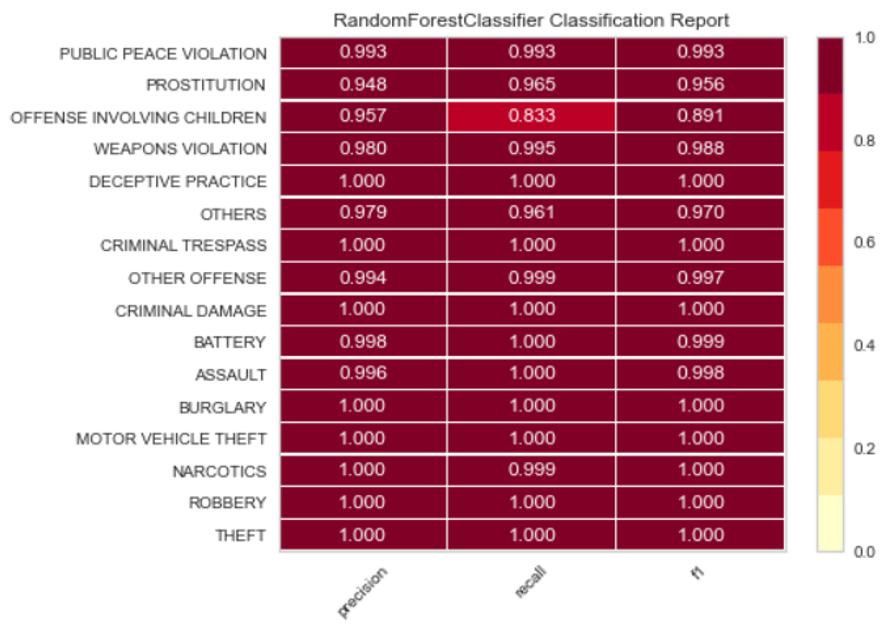
Ta có thể thấy hầu hết các tập dữ liệu riêng biệt đều được dự đoán đúng. Tuy nhiên, có một vài tập riêng có một lượng nhỏ dự đoán sai. Ví như tập Offense Involving Children mang giá trị Precision, Recall và F1\_Score lần lượt là 0.96, 0.83 và 0.89 với tổng số dữ liệu kiểm được (support) là 132. Điều này mang ý nghĩa là có các phần tử được dự báo không chính xác gây giảm độ chính xác của tập dữ liệu này.

Diễn giải kĩ hơn, ta có Offense Involving Children ở lớp thứ 14 (nhãn 14). Quay trở lại ma trận nhầm lẫn được tính toán ở trên, ta thấy:



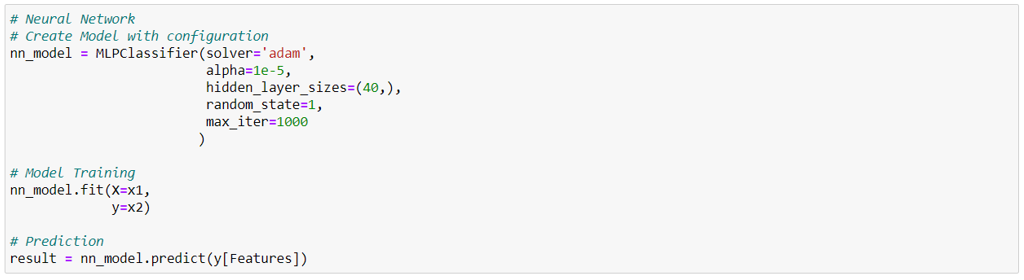
Như vậy trong tổng số thanh phần (132 giá trị) có 110 giá trị được phân lớp đúng và 22 giá trị bị phân lớp sai. Tỉ lệ điểm được phân lớp đúng:

110/132 = 0.83 = giá trị Recall tính được như báo cáo



Đây là mô hình được tạo trên bản báo cáo phân loại, với các màu sắc từ nhạt tới đậm biểu thị cho sự tăng tiến chính xác của tập dữ liệu kiểm được. Có thể thấy tập dữ liệu đa phần mang độ chính xác rất cao. Tuy nhiên vẫn còn tập dữ liệu riêng mang độ chính xác thấp hơn mặt bằng chung (Offense Involving Children) với tỉ lệ giá trị bị phân lớp sai là xấp xỉ 17% (22/132 giá trị).

#### Neural Network:



Để tạo mô hình Neural Network, chúng ta sẽ sử dụng hàm **MLPClassifier** của thư viện **sklearn** để khởi tạo mô hình. Các thông số của hàm khởi tạo như sau:

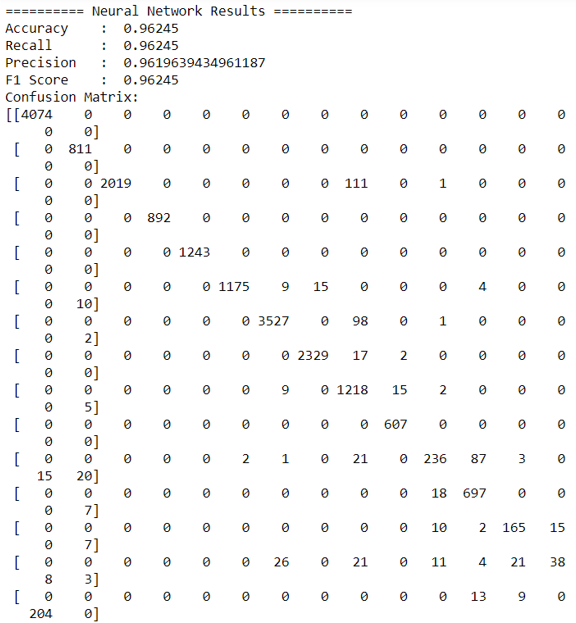
* + **Solver** (bao gồm các giá trị ‘lbfgs’, ‘sgd’, ‘adam’, mặc định là ‘adam’): Bộ giải để tối ưu hóa trọng số. ‘Lbfgs’ là một trình tối ưu hóa có liên quan đến phương pháp quasi-Newton.‘Sgd’ đề cập đến việc giảm độ dốc ngẫu nhiên.‘Adam’ đề cập đến một trình tối ưu hóa dựa trên độ dốc ngẫu nhiên được đề xuất bởi Kingma, Diederik và Jimmy Ba. Với các dữ liệu lớn (khoảng vài ngàn mẫu dữ liệu huấn luyện trở lên), thì phương pháp mặc định hoạt động tương đối tốt, về cả thời gian huấn luyện lẫn các thông số.
  + **Alpha** (kiểu giá trị số thực float, mặc định là 0.0001): Tham số hình phạt L2 (thuật ngữ chính quy). Đây là tham số để kiểm soát tốc độ học của mô hình. Giá trị ở đây là 1e-5.
  + **Hidden\_layer\_sizes** (số bộ, độ dài = số lớp -2, mặc định (100,): Phần tử thứ i đại diện cho số lượng tế bào thần kinh trong lớp ẩn thứ i. Ở đây giá trị là (40,).
  + **Random\_state** (kiểu số nguyên int, mặc định là 1): Xác định việc tạo số ngẫu nhiên cho trọng số và khởi tạo sai lệch, phân tách tập huấn luyện-kiểm tra nếu dừng sớm được sử dụng và lấy mẫu hàng loạt khi solver = ‘sgd’ hoặc ‘adam’. Truyền một int cho kết quả có thể lặp lại qua nhiều lệnh gọi hàm. Ở đây giá trị là 1.
  + **Max\_iter** (kiểu số nguyên int, mặc định là 100): Số lần lặp tối đa. Solver lặp lại cho đến khi hội tụ hoặc đạt tới số lần lặp này. Đối với solver ngẫu nhiên (‘sgd’,‘ adam’), lưu ý rằng điều này xác định số lượng epochs (số lần mỗi điểm dữ liệu sẽ được sử dụng), chứ không phải số bước gradient. Ở đây giá trị là 1000).

Sau đó, chúng ta sẽ tiến hành đào tạo mô hình trên tập huấn luyện và thực hiện các dự đoán trên tập kiểm tra.



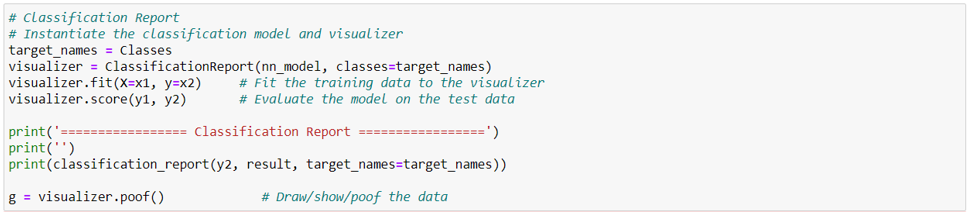
Tương tự như trên thuật toán Random Forest, ta cũng dùng ma trận nhầm lẫn (Confussion Matrix) để đánh giá độ chính xác mà thuật toán này mang lại trên tập dữ liệu.

Sau đó, chúng ta sẽ sử dụng ma trận nhầm lẫn để đánh giá đô chính xác của mô hình.



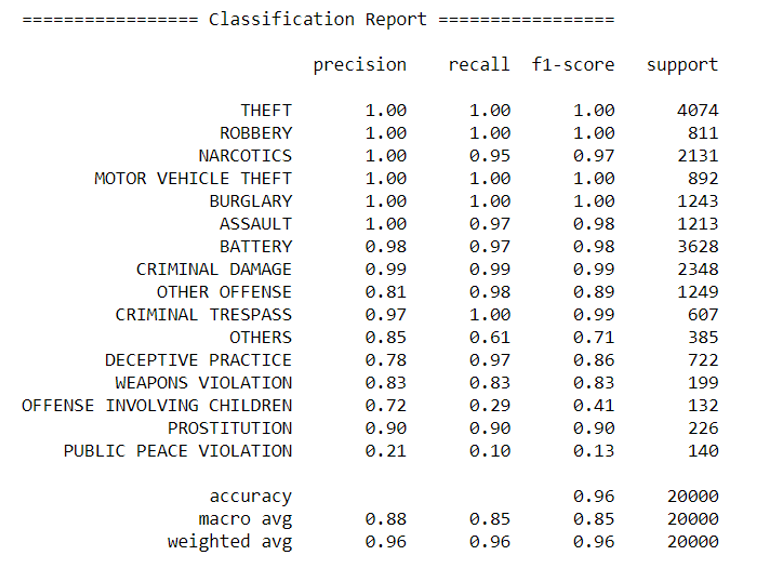
Như ta quan sát, độ chính xác mà thuật toán Neural Network mang lại là thấp hơn thuật toán Random Forest. Điều này có thể nhận thấy qua phần trăm độ chính xác chênh lệch khá nhiều (99.75% đối với thuật toán Random Forest và 96.25% đối với thuật toán Neural Network)

Rõ nhất là đối với lớp thứ 14 (Offense Involving Children), dữ liệu được phân lớp đúng chỉ có 38 giá trị với tỉ lệ là 29%



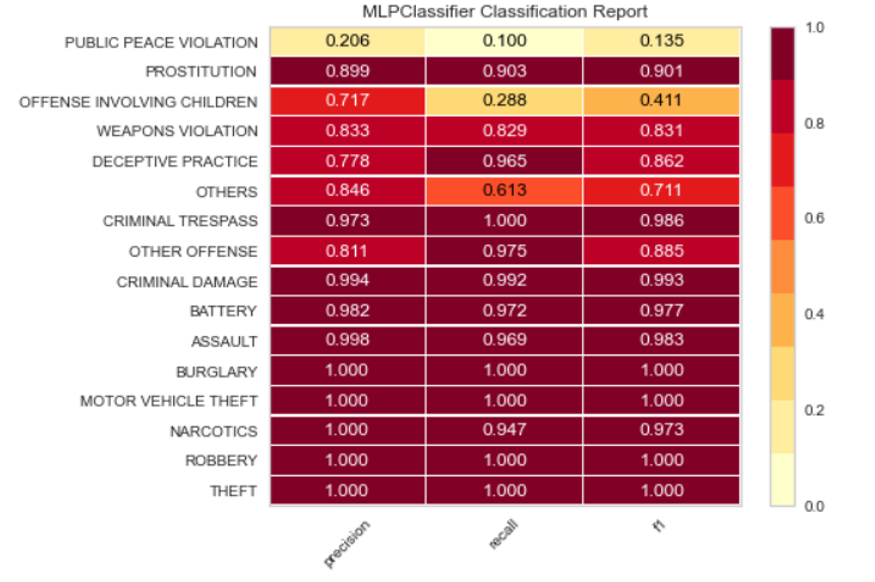
Sau đó, chúng ta sẽ tiến hành tạo báo cáo phân loại cho mô hình để tiện đánh giá mô hình.

Kết quả thu được như sau:



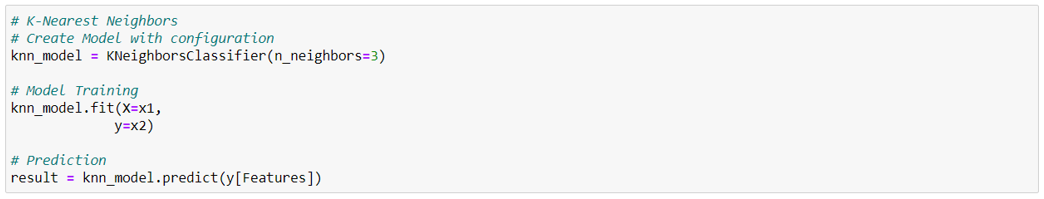
Đối với thuật toán này, độ chính xác của dữ liệu có sự chênh lệch lớn với độ chính xác cao nhất cho một phân lớp riêng là 100% và thấp nhất là 21%. Điều này cho thấy thuật toán này không phù hợp với bộ dữ liệu.

Mô hình độ chính xác và các giá trị liên quan được hiển thị như sau:



Nhìn vào biểu đồ này, ta có thể thấy rõ độ chênh lệch dựa vào các phân bố màu sắc.

#### K – Nearest Neighbors:



Để tạo mô hình K - Nearest Neighbors, chúng ta sẽ sử dụng hàm **KNeighborsClassifier** của thư viện **sklearn** để khởi tạo mô hình. Các thông số của hàm khởi tạo như sau:

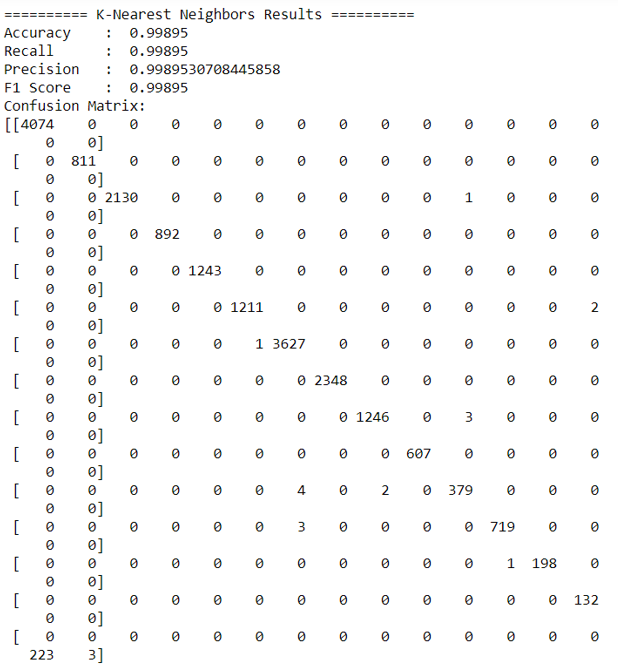
* + **N\_neighbors** (kiểu số nguyên int, mặc định là 5): Số nút dữ liệu gần nhất sẽ được sử dụng để dự đoán nhãn cho một điểm dữ liệu mới. Ở đây giá trị là 3.

Sau đó, chúng ta sẽ tiến hành đào tạo mô hình trên tập huấn luyện và thực hiện các dự đoán trên tập kiểm tra.



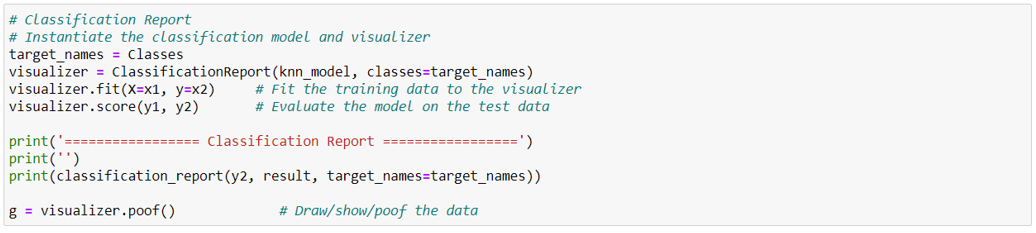
Tương tự như trên thuật toán Ran dom Forest, ta cũng dùng ma trận nhầm lẫn (Confussion Matrix) để đánh giá độ chính xác mà thuật toán này mang lại trên tập dữ liệu.

Sau đó, chúng ta sẽ sử dụng ma trận nhầm lẫn để đánh giá đô chính xác của mô hình.



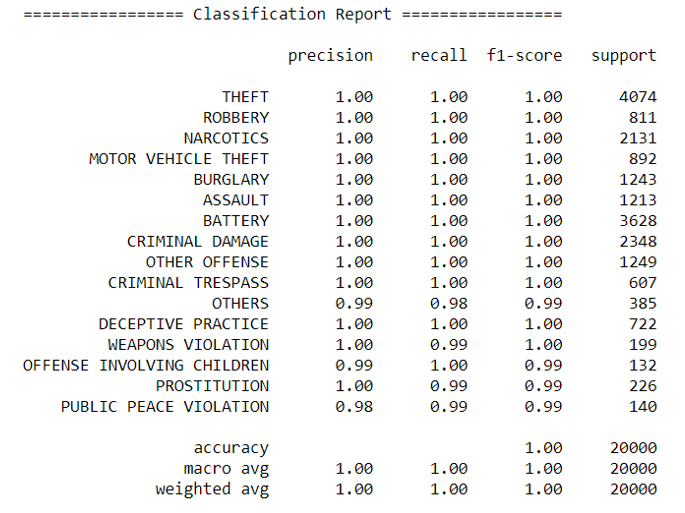
Như ta quan sát, độ chính xác mà thuật toán K - Nearest Neighbors mang lại là cao hơn thuật toán Forest, tuy độ chính xác chênh lệch giữa hai thuật toán là không đáng kể ( ~0.002)

Rõ nhất là đối với lớp thứ 14 (Offense Involving Children), dữ liệu được phân lớp đúng là 132 với tỉ lệ là 100%.



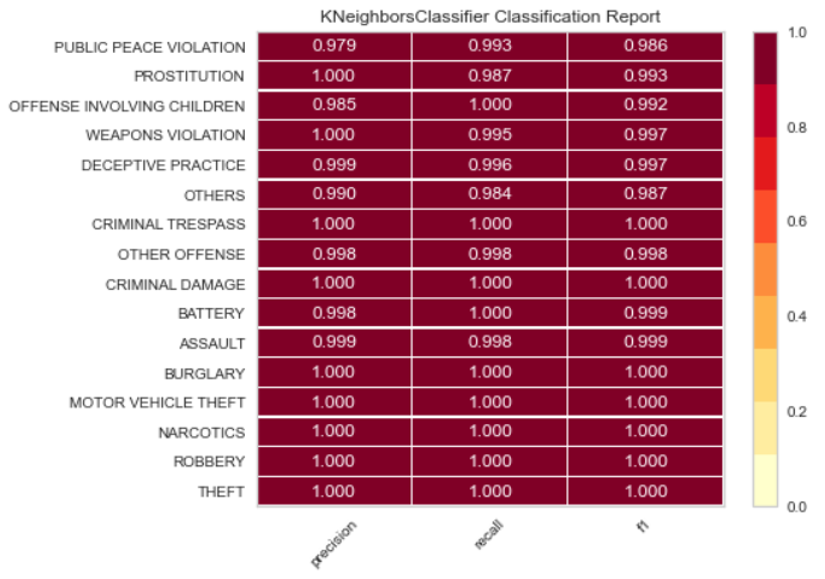
Sau đó, chúng ta sẽ tiến hành tạo báo cáo phân loại cho mô hình để tiện đánh giá mô hình.

Kết quả báo cáo phân loại như sau:



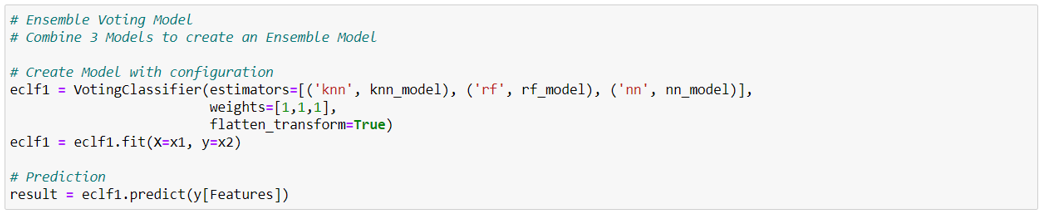
Với kết quả tỉ lệ phân lớp thấp nhất là 0.98 tức 98%, ta có thấy kết luận thuật toán này cho ra kết quả dự đoán gần như là chính xác 100%.

Phản ánh kết quả đạt được lên biểu đồ:



Dễ dàng nhận thấy mô hình mang lại kết quả phân lớp tại mức màu cao nhất của bảng màu phân lớp. Như vậy, sử dụng thuật toán K – Nearest Neighbors sẽ đem lại tập dữ liệu được dự đoán chính xác hơn.

#### Ensemble Voting:



Để tạo mô hình Ensemble Voting, chúng ta sẽ sử dụng hàm **VotingClassifier** của thư viện **sklearn** để khởi tạo mô hình. Các thông số của hàm khởi tạo như sau:

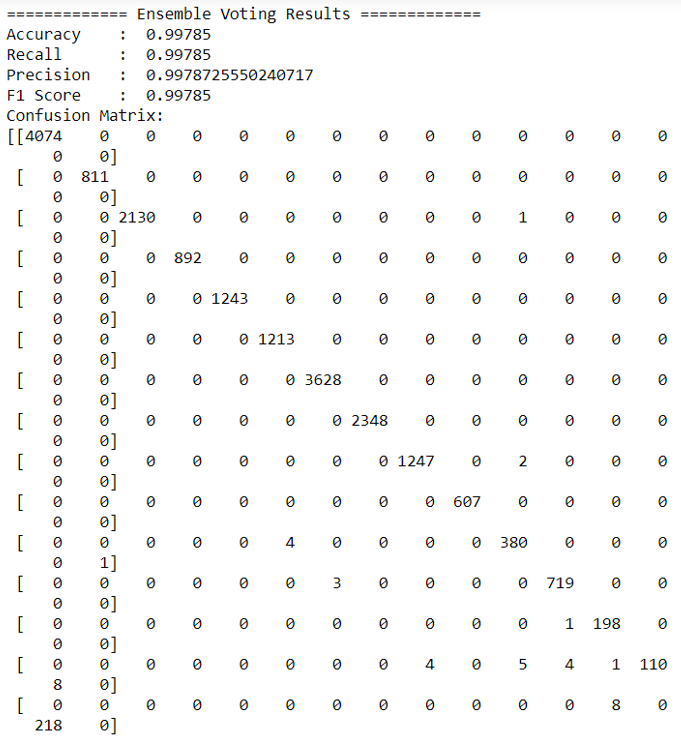
* + **Estimators** (truyền vào danh sách các mô hình sẽ được sử dụng để bầu chọn): dùng để ánh xạ các bản sao với các mô hình gốc.
  + **Weights** (truyền vào mảng có kích thước tương ứng với số classifier, mặc định là không có): Trình tự các trọng số (float hoặc int) để cân nhắc sự xuất hiện của các nhãn lớp dự đoán (bỏ phiếu cứng) hoặc xác suất của lớp trước khi tính trung bình (bỏ phiếu mềm). Sử dụng trọng lượng bằng nhau nếu không đặt thông số này. Ở đây có thể thấy trọng số của 3 classifier gốc là bằng nhau và bằng 1.
  + **Flatten\_transform** (kiểu bool, mặc định là True): Ảnh hưởng đến đầu ra biến đổi. Nếu flatten\_transform = True, phương pháp biết đổi sẽ trả về kết quả là một ma trận với hình dạng (n\_samples, n\_classifiers \* n\_classes). Nếu flatten\_transform = False, nó sẽ trả về kết quả (n\_classifiers, n\_samples, n\_classes). Chỉ ảnh hưởng đến bỏ phiếu mềm.

Sau đó, chúng ta sẽ tiến hành đào tạo mô hình trên tập huấn luyện và thực hiện các dự đoán trên tập kiểm tra.

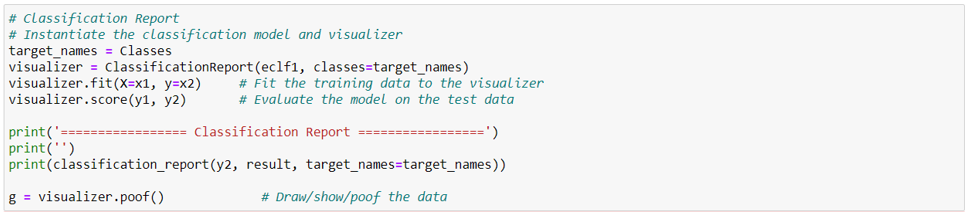


Tương tự như các thuật toán trên, ta dung các giá trị Precision, Recall và F1 – Score để dự báo độ chính xác của tập dữ liệu thu được.

Sau đó, chúng ta sẽ sử dụng ma trận nhầm lẫn để đánh giá đô chính xác của mô hình.

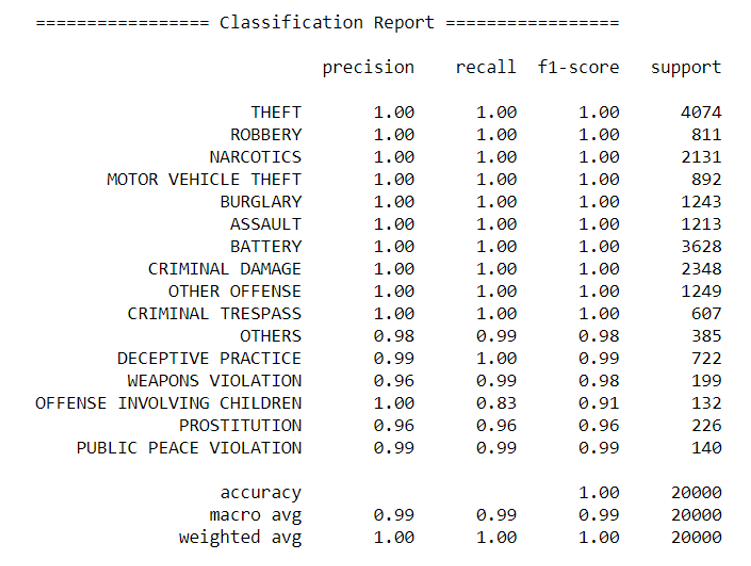


Dựa vào kết quả của ma trận nhầm lẫn, ta thấy được độ chính xác của tập dữ liệu khi sử dụng thuật toán Ensemble Voting là cao hơn thuật toán Random Forest nhưng vẫn chưa có độ chính xác bằng thuật toán K – Nearest Neighbors (0.9975 < 0.9979 < 0.999).



Sau đó, chúng ta sẽ tiến hành tạo báo cáo phân loại cho mô hình để tiện đánh giá mô hình.

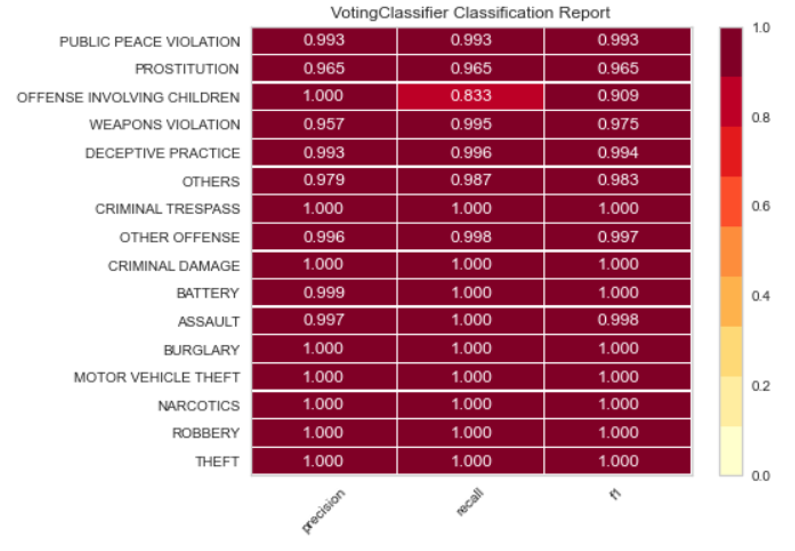
Kết quả được hiển thị như sau:



Nhìn vào số liệu này, ta có thể thấy sự tương đồng giữa kết quả của thuật toán Random Forest và thuật toán này.

Đây cũng là một trong các thuật toán đưa ra được tập dữ liệu kiểm thử chính xác đối với tập dữ liệu này.

Mô hình hiển thị theo phân cách màu sắc như sau:



Dễ dàng thấy tập dữ liệu dự báo này mang độ chính xác không kém gì thuật toán Random Forest. Do độ chính xác chênh lệch là rất thấp nên kết quả hiển thị được tường đối giống nhau.

Như vậy, thuật toán này cũng phù hợp trong việc dự báo cho tập dữ liệu này.

# KẾT QUẢ:

### So sánh:

Các thuật toán đều có điểm mạnh và điểm yếu, song, tất cả đều phải thử nghiệm và tìm hiểu để đưa ra các lựa chọn phù hợp nhất cho tập dữ liệu cần dự báo.

Đối với thuật toán **Random Forest**, một tập dữ liệu với độ chính xác cao đã được xây dựng. Với tỉ lệ phân lớp sai là thấp với 17% là thấp nhất đối với một thuộc tính riêng biệt. Trong 132 giá trị cần phân lớp chỉ tồn tại 22 giá trị bị phân lớp sai. Điều này cho thấy sự tối ưu của thuật toán đối với các bộ dữ liệu lớn.

Đối với thuật toán **Neural Network**, một tập dữ liệu tương đối đã được xây dựng. Với độ chính xác giữa mức cao nhất và thấp nhất có sự chênh lệch lớn, thuật toán này không phải là thuật toán lý tưởng cho bộ dữ liệu này. Đồng thời, phần trăm của dữ liệu bị phân lớp sai đối với một thuộc tính riêng biệt là cao với 71% sai lệch. Do sự sai lệch đó, độ chính xác của toàn bộ tập dữ liệu cũng bị giảm đi rất nhiều. Rõ ràng nhất là thuộc tính **Public Peace Violation** khi được dự báo bằng thuật toán K -Nearest Neighbors thì độ chính xác là 97.9%, trong khi được dự báo bằng thuật toán Neural Network độ chính xác chỉ có 20,6% với tỉ lệ phân lớp đúng là 10%.

Đối với thuật toán **K – Nearest Neighbors**, một tập dữ liệu với độ chính xác rất cao đã được xây dựng. Với tỉ lệ phân lớp sai là thấp với 1.56% là thấp nhất đối với một thuộc tính riêng biệt. Trong 385 giá trị cần phân lớp chỉ tồn tại 6 giá trị bị phân lớp sai. Điều này cho thấy sự tối ưu của thuật toán đối với các bộ dữ liệu lớn.

Đối với thuật toán **Ensemble Voting**, một tập dữ liệu với độ chính xác cao đã được xây dựng. Với tỉ lệ phân lớp sai là thấp với 17% là thấp nhất đối với một thuộc tính riêng biệt. Trong 132 giá trị cần phân lớp chỉ tồn tại 22 giá trị bị phân lớp sai. Điều này cho thấy sự tối ưu của thuật toán đối với các bộ dữ liệu lớn. Đối với kết quả này, các thông số không khác là bao khi so sánh với thuật toán Random Forest, do đó, trong bộ dữ liệu này, ta có thể nói độ chính xác của hai thuật toán này là ngang nhau.

### Đánh giá:

Có rất nhiều thuật toán có thể sử dụng để xây dựng bộ dữ liệu kiểm thử. Tuy nhiên, không phải thuật toán nào cũng phù hợp và đưa ra được bộ dữ liệu dự đoán chính xác. Do đó, việc sử dụng Ma trận nhầm lẫn (Confussion Matrix) chính là giải pháp được đưa ra để kiểm tra độ chính xác (Precision), tỉ lệ phủ (Recall) của mô hình thuật toán cũng như tính chính xác trên toàn tập dữ liệu (F1 -Score).

Chỉ xét trên bộ dữ liệu tội phạm tại Chicago (Crime in Chicago) này, chúng em đã sử dụng tổng cộng 4 thuật toán khác nhau để đưa ra tập dữ liệu kiểm thử. Tuy nhiên, không phải tất cả đều phù hợp.

Thông qua quá trinh sử dụng các thuật toán để dự báo cho tập dữ liệu kiểm thử, nhóm chúng em đã tìm hiểu được phương pahsp đánh giá cũng như có thể tính toán mức độ chính xác nào là tốt đối với tập dữ liệu kiểm thử.

Qua quá trình đó, nhóm chúng em cũng rút ra được ba thuật toán phù hợp với tập dữ liệu trên là thuật toán Random Forest, K – Nearest Neighbors và Ensemble Voting. Trong đó, phù hợp nhất là thuật toán K - Nearest Neighbors.

# KẾT LUẬN:

### Ưu điểm:

Chỉ đối với bộ dữ liệu này, các thuật toán được áp dụng gồm những thuật toán từ cơ bản như Random Forest, Neural Network và K – Nearest Neighbors cho tới thuật toán ít thông dụng hơn như Ensemble Voting đều cho thấy mức độ phù hợp và ưu điểm của nó trong việc dự báo tập dữ liệu.

Đối với thuật toán **Random Forest**, ưu điểm của nó là tính chính xác và mạnh mẽ do số cây quyết định tham gia vào quá trình này. Random Forest không bị vấn đề overfitting. Lý do chính là nó mất trung bình của tất cả các dự đoán, trong đó hủy bỏ những thành kiến. Thuật toán có thể được sử dụng trong cả hai vấn đề phân loại và hồi quy. Random Forest cũng có thể xử lý các giá trị còn thiếu. Có hai cách để xử lý các giá trị này: sử dụng các giá trị trung bình để thay thế các biến liên tục và tính toán mức trung bình gần kề của các giá trị bị thiếu. Bạn có thể nhận được tầm quan trọng của tính năng tương đối, giúp chọn các tính năng đóng góp nhiều nhất cho trình phân loại. Thuật toán này là thuật toán mà bất kỳ ai cũng có thể thử nghiệm trên bộ dữ liệu của mình với cách thực hiện đơn giản.

Đối với thuật toán **Neural Network**, ưu điểm của nó là khả năng tự học và điều chỉnh các trọng số để kết quả tính toán phù hợp với thực tế mà không phụ thuộc vào ý kiến chủ quan. Thuật toán này được coi là công cụ mạnh để giải quyết các bài toán có tính phi tuyến, phức tạp và đặc biệt trong các trường hợp mà mối quan hệ giữa các quá trình không dễ thiết. Sử dụng thuật toán này còn có thể tăng tỉ lệ dữ liệu đầu vào hiệu dụng, tiết kiệm tài nguyên bộ nhớ cũng như tài nguyên cổng logic phần cứng.

Đối với thuật toán **K – Nearest Neighbors**, ưu điểm của nó là độ phức tạp tính toán của quá trình training nhỏ, việc dữ đoán cũng đơn giản và không cần giả sử gì về phân phối của các class.

Đối với thuật toán **Ensemble Voting**, ưu điểm của thuật toán này là hỗ trợ được cho các tập không đầy đủ dữ liệu, tập được đưa vào kiểm định mang lượng thông tin quá nhỏ hoặc lượng thông tin không đồng nhất. Ưu điểm lớn nhất của nó là tính kết hợp và tạo các mô hình con của tập dữ liệu. Từ đó đưa ra được kết quả dự báo có tính chính xác hơn.

### Hạn chế:

Đi kèm với những ưu điểm mà thuật toán đã được đưa ra, vẫn còn các yếu điểm mà thuật toán cần khắc phục để có thể phù hợp với nhiều dạng dữ liệu hơn.

Đối với **Random Forest**, thuật toán này chậm tạo dự đoán bởi vì nó có nhiều cây quyết định. Bất cứ khi nào nó đưa ra dự đoán, tất cả các cây trong rừng phải đưa ra dự đoán cho cùng một đầu vào cho trước và sau đó thực hiện bỏ phiếu trên đó. Toàn bộ quá trình này tốn thời gian. Mô hình khó hiểu hơn so với cây quyết định, nơi bạn có thể dễ dàng đưa ra quyết định bằng cách đi theo đường dẫn trong cây.

Đối với thuật toán **Neural Network**, thuật toán này có số bộ nhân và bộ cộng tăng lên tương ứng với số đường dữ liệu vào. Nếu có số lượng đường dữ liệu vào lớn sẽ ảnh hưởng đến yêu cầu về kích thước của vi mạch thực hiện. Ngoài ra, thuật toán này cũng khó xây dựng hàng đợi và đồng bộ dữ liệu đầu vào khi thực hiện trên nhiều lớp nơron. Số thanh ghi và flipflop tăng lên để lưu giữ giá trị khi tiết kiệm cổng logic, cùng với đó là chu kỳ một vòng tính toán lớn khiến bộ điều khiển trở nên phức tạp và khó kiểm soát. Điều đó khiến cho thời gian cần để thực hiện thuật toán lớn và chậm do khi dữ liệu vào phải tính toán nối tiếp.

Đối với thuật toán **K - Nearest Neighbors**, khi K nhỏ thuật toán này sẽ trở nên rất nhạy cảm với dữ liệu nhiễu. Đồng thời đây là một thuật toán mà mọi tính toán đều nằm ở khâu test. Trong đó việc tính khoảng cách tới *từng* điểm dữ liệu trong training set sẽ tốn rất nhiều thời gian, đặc biệt là với các cơ sở dữ liệu có số chiều lớn và có nhiều điểm dữ liệu. Với K càng lớn thì độ phức tạp cũng sẽ tăng lên. Ngoài ra, việc lưu toàn bộ dữ liệu trong bộ nhớ cũng ảnh hưởng tới hiệu năng của thuật toán.

Đối với thuật toán **Ensemble Voting**, việc lựa chọn các mô hình “yếu” để tạo ra mô hình “mạnh” là rất quan trọng. Việc chọn lựa các mô hình “yếu” này cũng làm tăng số lượng mô hình cần sử dụng lên và cần train các mô hình này theo những hướng khác nhau. Do đó, thời gian chạy thuật toán này sẽ cao và chậm do các mô hình “yếu” liên tục được thêm vào để củng cố tính chính xác của mô hình “mạnh” nhận được.

### Hướng phát triển:

Ngày nay, việc sử dụng dữ liệu vào các nền tảng thông tin và các hoạt động khác trong xã hội là rất quan trọng. Càng quan trọng hơn nữa khi bất kỳ ngành nghề nào cũng sở hữu cho mình một cơ sở dữ liệu khổng lồ và không bao giờ ngừng cập nhật và lớn mạnh. Do đó, việc quản lý và sử dụng nguồn dữ liệu đó một cách đúng và hiệu quả nhất luôn là điều mà con người tìm kiếm.

Khai thác dữ liệu được ngầm hiểu như một cách hiệu quả mà con người đang muốn phát triển đó. Việc sử dụng các thuật toán dự báo này sẽ được ra được tính chính xác của dữ liệu cùng với mô hình dự đoán có lý lẽ và cơ sở. Điều này đem lại lợi ích to lớn cho các ngành nghề khi họ có được một bộ dữ liệu sạch, đúng và có thể sử dụng.

Trong báo cáo trên, chúng em đã dùng tổng cộng 4 thuật toán. Kết quả dự đoán mô hình là có độ chính xác cao gồm 3 thuật toán và 1 thuật toán còn lại cho kết quả độ chính xác của dự đoán rất thấp. Dựa vào đó, ta cũng có thể xác định hướng phát triển của bộ dữ liệu thiên về hướng phân lớp hay gọm cụm thì tập dữ liệu sẽ hiệu quả hơn.

Để phát triển bộ dữ liệu này hơn nữa, chúng em cần tìm hiểu các thêm các thuật toán thiên về gom cụm để củng cố độ chính xác và độ sai lệch là do đâu. Điều này sẽ chỉ ra hướng phát triển và cách sử dụng bộ dữ liệu này một cách hợp lý nhất.

Quay trở lại bản báo cáo, sau quá trình áp dụng các thuật toán vào tập dữ liệu, chúng em đã thu được kết quả thuật toán phù hợp nhất với bộ dữ liệu là thuật toán K – Nearest Neighbors. Đối với thuật toán này, các ứng dụng chúng em có thể phát triển dự trên tính toán của nó là Dựa vào dự báo của nó đưa ra các loại tội phậm thường xuyên xảy ra nhất. Kết hợp với dữ liệu tọa độ (X, Y) xảy ra vụ án, việc đưa ra các cảnh báo cho người dân gần đó về một hình thức tội phạm đang trở nên phổ biến là hoàn toàn cần thiết và phù hợp.

Ngoài ra, một ứng dụng phù hợp với thuật toán này là một ứng dụng thông báo đến người dân các tội phạm liên quan đến phương thức phạm tội phổ biến tại nơi họ ở (dựa trên tọa độ X, Y) mà chưa bị bắt (Dựa trên Arrested). Hơn cả điều đó, các phương pháp dự báo này còn có thể đưa ra các khả năng loại tội phạm nào sẽ bùng nổ cùng với tỉ lệ xảy ra khả năng đó.

Như vậy, việc đưa ra được các Prediction này sẽ giúp người dân tìm ra cách phòng tránh, hoặc chí ít có thể có chuẩn bị trước các công tác bảo vệ phù hợp để bảo vệ chính bản thân họ. Việc tạo các khả năng này cũng nâng cao ý thức tự bảo vệ bản thân, tính cảnh giác và giảm sự chủ quan ở người dân. Đây là điều cần thiết trước tình hình các loại tội phạm không ngừng tăng cao tại Chicago.

# BẢNG PHÂN CÔNG:

|  |  |  |  |
| --- | --- | --- | --- |
| TÊN | MSSV | PHÂN CÔNG | HOÀN THÀNH |
| Hà Trọng Nhân | 17520836 | Chạy các thuật toán trên nền tảng Python  Viết báo cáo mục các thuật toán, lý do lựa chọn và các bước chạy thuật toán. | 100% |
| Phan Thị Long Huệ | 17520530 | Chạy các thuật toán trên nền tảng Python  Viết báo cáo nêu bài toán cần giải quyết và các kỹ thuật tiền xử lý dữ liệu. | 100% |
| Khổng Minh Quốc | 17520954 | Chạy các thuật toán trên nền tảng Python  Viết báo cáo mục giới thiệu đề tài, lý do chọn lựa đề tài và mô tả các thuộc tính bộ dữ liệu. | 100% |
| Nguyễn Thị Ngọc Hà | 17520421 | Chạy các thuật toán trên nền tảng Python  Viết báo cáo giải thích các độ đo của kết quả thuật toán, mục kết quả so sánh, đánh giá và kết luận. | 100% |

# BẢNG ĐÁNH GIÁ CHÉO:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| TÊN | Hà Trọng Nhân | Phan Thị Long Huệ | Khổng Minh Quốc | Nguyễn Thị Ngọc Hà |
| Hà Trọng Nhân |  | 100% | 100% | 100% |
| Phan Thị Long Huệ | 100% |  | 100% | 100% |
| Khổng Minh Quốc | 100% | 100% |  | 100% |
| Nguyễn Thị Ngọc Hà | 100% | 100% | 100% |  |

# TÀI LIỆU THAM KHẢO:

1. <https://viblo.asia/p/phan-lop-bang-random-forests-trong-python-djeZ1D2QKWz>
2. <https://stackoverrun.com/vi/q/8460472>
3. <https://scikit-learn.org/stable/auto_examples/model_selection/plot_confusion_matrix.html>
4. <https://machinelearningcoban.com/2017/08/31/evaluation/#-confusion-matrix>
5. <https://machinelearningcoban.com/2017/01/08/knn/>
6. <https://machinelearningcoban.com/2016/12/26/introduce/>
7. <https://machinelearningcoban.com/2017/01/21/perceptron/>
8. <https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.VotingClassifier.html>
9. <https://codetudau.com/neural-network-va-deep-learning-la-gi/index.html>
10. <https://scikit-learn.org/stable/modules/neural_networks_supervised.html>