**ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

**KHOA HỆ THỐNG THÔNG TIN**

****

**BÁO CÁO ĐỒ ÁN MÔN HỌC**

**KHAI THÁC DỮ LIỆU**

**ĐỀ TÀI**

**DỰ ĐOÁN MỨC ĐỘ NGHIÊM TRỌNG CỦA TAI NẠN XE HƠI**

**Nhóm sinh viên thực hiện:**

**1. Lê Thị Thanh Hằng MSSV: 21520222**

**2. Ngô Tất Tố MSSV: 21520484**

**3. Nguyễn Nhật Phương Huy MSSV: 21522156**

**4. Lê Xuân Thạch MSSV: 21521421**

**TP. HỒ CHÍ MINH, NĂM 2024**

**NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN**

*……., ngày……...tháng……năm 2023*

**Người nhận xét**

*(Ký tên và ghi rõ họ tên****)***

**MỤC LỤC**

[**CHƯƠNG 1: GIỚI THIỆU ĐỀ TÀI 3**](#_Toc167695653)

[1.1. Lý do chọn đề tài 3](#_Toc167695654)

[1.2. Giới thiệu về đề tài 3](#_Toc167695655)

[**1.2.1. Thông tin của dataset 3**](#_Toc167695656)

[**1.2.2. Tác giả 3**](#_Toc167695657)

[**1.2.3. Số cột, số dòng, khoảng thời gian 3**](#_Toc167695658)

[**1.2.4. Nguồn dataset 4**](#_Toc167695659)

[**1.2.5. Thông tin chi tiết của các thuộc tính 4**](#_Toc167695660)

[1.3. Thống kê mô tả và trực quan hóa dữ liệu (EDA) 5](#_Toc167695661)

[**1.3.1. Thống kê mô tả 5**](#_Toc167695662)

[**1.3.2. Trực quan hóa dữ liệu 7**](#_Toc167695663)

[1.4. Mô tả bài toán 10](#_Toc167695664)

[1.5. Công cụ thực hiện 10](#_Toc167695665)

[**CHƯƠNG 2: XỬ LÝ TIỀN DỮ LIỆU 11**](#_Toc167695666)

[2.1. Thêm các thuộc tính 11](#_Toc167695667)

[2.2. Kiểm tra độ tương quan giữa các thuộc tính 11](#_Toc167695668)

[2.3. Lựa chọn thuộc tính 13](#_Toc167695669)

[2.4. Kiểm tra dữ liệu null 14](#_Toc167695670)

[2.5. Loại bỏ trùng lặp 16](#_Toc167695671)

[2.6. Kiểm tra phương sai của các thuộc tính 17](#_Toc167695672)

[**CHƯƠNG 3: ÁP DỤNG CÁC THUẬT TOÁN KHAI THÁC 27**](#_Toc167695673)

[3.1. Chia dữ liệu 27](#_Toc167695674)

[3.2. KNN 27](#_Toc167695675)

[**3.2.1. Giới thiệu thuật toán 27**](#_Toc167695676)

[**3.2.2. Lý do chọn thuật toán 28**](#_Toc167695677)

[**3.2.3. Xây dựng và dự đoán với mô hình 28**](#_Toc167695678)

[3.3. Decision tree 31](#_Toc167695679)

[**3.3.1. Giới thiệu thuật toán Decision tree 31**](#_Toc167695680)

[**3.3.2. Lý do chọn thuật toán 32**](#_Toc167695681)

[**3.3.3. Xây dựng và dự đoán với mô hình 32**](#_Toc167695682)

[3.4. Random forest 37](#_Toc167695683)

[**3.4.1. Giới thiệu thuật toán Random Forest (RF) 37**](#_Toc167695684)

[**3.4.2. Lý do chọn thuật toán: 37**](#_Toc167695685)

[**3.4.3. Xây dựng và dự đoán với mô hình RF 38**](#_Toc167695686)

[3.5. Naïve Bayes 42](#_Toc167695687)

[**3.5.1. Giới thiệu thuật toán Navie Bayes 42**](#_Toc167695690)

[**3.5.2. Lý do chọn thuật toán 43**](#_Toc167695691)

[**3.5.3. Xây dựng và dự đoán với mô hình 43**](#_Toc167695692)

[**CHƯƠNG 4: PHƯƠNG PHÁP ĐÁNH GIÁ VÀ DỰ ĐOÁN 48**](#_Toc167695693)

[4.1. Đánh giá các thuật toán bằng Accuracy 48](#_Toc167695694)

[**4.1.1. Giới thiệu phương pháp đánh giá 48**](#_Toc167695695)

[**4.1.2. Lý do lựa chọn phương pháp 48**](#_Toc167695696)

[**4.1.3. Tính toán 49**](#_Toc167695697)

[4.2. Phương pháp đánh giá F1 – score 50](#_Toc167695698)

[**4.2.1. Giới thiệu phương pháp đánh giá F1 – score 50**](#_Toc167695699)

[**4.2.2. Lí do chọn phương pháp 52**](#_Toc167695700)

[**4.2.3. Tính toán Độ chính xác Macro, Recall và F1 – score 53**](#_Toc167695701)

[4.3. Phương pháp ROC 55](#_Toc167695702)

[**4.3.1. Giới thiệu 55**](#_Toc167695703)

[**4.3.2. Lý do chọn phương pháp 57**](#_Toc167695704)

[**4.3.3. Tính toán 58**](#_Toc167695705)

[4.4. Dự đoán 63](#_Toc167695706)

[**CHƯƠNG 5: TỔNG KẾT 67**](#_Toc167695707)

[5.1. Ưu điểm 67](#_Toc167695708)

[5.2. Hạn chế 67](#_Toc167695709)

[5.3. Kết quả đạt được 67](#_Toc167695710)

[5.4. Định hướng phát triển 67](#_Toc167695711)

[**DANH MỤC THAM KHẢO 68**](#_Toc167695712)

# GIỚI THIỆU ĐỀ TÀI

## Lý do chọn đề tài

Tai nạn giao thông là vấn đề nhức nhối trên toàn thế giới, gây ra hậu quả vô cùng nghiêm trọng về người và của. Việc phân tích dữ liệu giao thông đóng vai trò then chốt trong việc phòng ngừa và giảm thiểu tai nạn, góp phần bảo vệ an toàn cho người tham gia giao thông.

Phân tích dữ liệu tai nạn giao thông giúp xác định các yếu tố dẫn đến tai nạn như: hành vi của người lái xe, điều kiện đường sá, loại xe, yếu tố môi trường,.. nhờ đó, chúng ta có thể đưa ra các biện pháp phòng ngừa hiệu quả như:

* Thực thi nghiêm luật giao thông
* Cải thiện cơ sở hạ tầng
* Kiểm soát chất lượng xe cộ
* Nâng cao nhận thức cộng đồng

Hiểu ra nguyên nhân tai nạn là bước đầu tiên để xây dựng các biện pháp phòng ngừa hiệu quả. Do vậy, việc phân tích dữ liệu tai nạn giao thông cần được quan tâm và đẩy mạnh. Đo cũng chính là lý do chúng tôi lựa chọn đề tài khai thác dữ liệu “Car Accidents”.

## Giới thiệu về đề tài

### Thông tin của dataset

Tên dataset: Car Accident Dataset

* Tập dữ liệu này cung cấp hồ sơ chi tiết về các vụ tai nạn đường bộ xảy ra trong tháng 1 năm 2021. Nó bao gồm các thông tin như ngày xảy ra tai nạn, ngày trong tuần, khu vực kiểm soát giao lộ, mức độ nghiêm trọng của tai nạn, tọa độ địa lý, điều kiện ánh sáng và thời tiết, chi tiết về phương tiện, v.v. Dữ liệu này có giá trị trong việc phân tích và hiểu rõ các yếu tố góp phần gây ra tai nạn giao thông đường bộ ở khu vực đô thị này, hỗ trợ phát triển các chiến lược cải thiện an toàn đường bộ.
* Cập nhật lần cuối: 01/2024

### Tác giả

* Tên tác giả: SAHER MUHAMED
* Tác giả là Nhà khoa học dữ liệu ở Ai Cập.

### Số cột, số dòng, khoảng thời gian

* Dữ liệu bao gồm: 21 cột, 307973 hồ sơ tại nan.
* Sau khi tiền xử lý: 16 cột, 300485 hồ sơ tai nạn.
* Sau khi mã hóa: 16 cột, 171016 hồ sơ tai nạn.

### Nguồn dataset

* Link dataset: <https://www.kaggle.com/datasets/nextmillionaire/car-accident-dataset>

### Thông tin chi tiết của các thuộc tính

|  |  |  |  |
| --- | --- | --- | --- |
| STT | Thuộc tính | Ý nghĩa | Kiểu dữ liệu |
| 1 | Accident\_Index | Một mã định danh duy nhất cho mỗi hồ sơ vụ tai nạn. | String |
| 2 | Accident Date | Ngày xảy ra tai nạn (định dạng: DD/MM/YYYY). | Date |
| 3 | Day\_of\_Week | Ngày trong tuần xảy ra tai nạn. | String |
| 4 | Junction\_Control | Mô tả loại kiểm soát nút giao tại địa điểm xảy ra tai nạn (ví dụ: "Nhường đường hoặc không kiểm soát"). | String |
| 5 | Junction\_Detail | Cung cấp thông tin chi tiết bổ sung về giao lộ nơi xảy ra tai nạn (ví dụ: "T hoặc giao lộ so le"). | String |
| 6 | Accident\_Severity | Cho biết mức độ nghiêm trọng của vụ tai nạn (ví dụ: "Nghiêm trọng"). | String |
| 7 | Latitude | Vĩ độ địa lý của nơi xảy ra tai nạn. | Float |
| 8 | Light\_Conditions | Mô tả điều kiện ánh sáng tại thời điểm xảy ra tai nạn (ví dụ: "Ánh sáng ban ngày"). | String |
| 9 | Local\_Authority\_(District) | Chính quyền địa phương nơi xảy ra vụ tai nạn. | String |
| 10 | Carriageway\_Hazards | Mô tả bất kỳ mối nguy hiểm nào hiện diện trên đường vào thời điểm xảy ra tai nạn (ví dụ: "Không"). | String |
| 11 | Longitude | Kinh độ địa lý của nơi xảy ra tai nạn. | Float |
| 12 | Number\_of\_Casualties | Tổng số người bị thương trong vụ tai nạn. | Int |
| 13 | Number\_of\_Vehicles | Tổng số phương tiện tham gia vụ tai nạn. | Int |
| 14 | Police\_Force | Lực lượng công an xử lý vụ tai nạn. | String |
| 15 | Road\_Surface\_Conditions | Mô tả tình trạng mặt đường tại thời điểm xảy ra tai nạn (ví dụ: "Khô"). | String |
| 16 | Road\_Type | Chỉ định loại đường nơi xảy ra tai nạn (ví dụ: "Đường một chiều"). | String |
| 17 | Speed\_limit | Giới hạn tốc độ áp dụng cho đoạn đường xảy ra tai nạn. | Int |
| 18 | Time | Thời gian trong ngày xảy ra tai nạn (định dạng: HH:MM). | Time |
| 19 | Urban\_or\_Rural\_Area | Cho biết tai nạn xảy ra ở khu vực thành thị hay nông thôn. | String |
| 20 | Weather\_Conditions | Mô tả điều kiện thời tiết tại thời điểm xảy ra tai nạn (ví dụ: "Tốt, không có gió lớn"). | String |
| 21 | Vehicle\_Type | Chỉ định loại phương tiện liên quan đến vụ tai nạn (ví dụ: "Ô tô", "Taxi/xe thuê riêng"). | String |

## Thống kê mô tả và trực quan hóa dữ liệu (EDA)

### Thống kê mô tả

Tính toán các giá trị: **Count, Min, Max, Mean, Median, Quantile, Range, Mode and Variance** của tập dữ liệu.

* Count (Số lượng): Số lượng quan sát hoặc mẫu trong tập dữ liệu. Đây là một chỉ số đơn giản nhất và cung cấp thông tin về kích thước của tập dữ liệu.
* Min (Giá trị nhỏ nhất): Giá trị nhỏ nhất trong tập dữ liệu. Đây là giá trị nhỏ nhất mà ta có thể tìm thấy trong tập dữ liệu.
* Max (Giá trị lớn nhất): Giá trị lớn nhất trong tập dữ liệu. Đây là giá trị lớn nhất mà ta có thể tìm thấy trong tập dữ liệu.
* Mean (Trung bình): Giá trị trung bình của tập dữ liệu, là tổng của tất cả các giá trị chia cho số lượng các giá trị. Nó cung cấp một sự đại diện trung tâm cho tập dữ liệu.
* Median (Trung vị): Giá trị ở giữa của tập dữ liệu khi được sắp xếp theo thứ tự tăng dần. Nó phản ánh giá trị trung tâm của tập dữ liệu mà không bị ảnh hưởng bởi các giá trị ngoại lệ.
* Quantile (Phân vị): Là giá trị chia tập dữ liệu thành các phần bằng nhau. Các phân vị phổ biến nhất bao gồm phân vị 25% (Q1), phân vị 50% (Median hay Q2), và phân vị 75% (Q3).
* Range (Phạm vi): Sự khác biệt giữa giá trị lớn nhất và giá trị nhỏ nhất trong tập dữ liệu. Nó cung cấp một ý tưởng về biến động của dữ liệu.
* Mode (Kiểu): Giá trị xuất hiện nhiều nhất trong tập dữ liệu. Nó cung cấp thông tin về xu hướng và phân phối của dữ liệu.
* Variance (Phương sai): Độ biến động của các giá trị so với giá trị trung bình. Nó đo lường mức độ phân tán của dữ liệu và cho biết độ lệch của dữ liệu so với trung bình.

**Cách thức:**

* Nhận thấy rằng bộ dữ liệu chứa hai loại dữ liệu số, là Int và Float nên tiến hành lọc các cột với loại dữ liệu Int hoặc Float để tính toán.
* Sử dụng phương thức Describe() để tính toán số liệu thống kê mô tả cho tất cả các cột với các loại dữ liệu của int hoặc float.

A screenshot of a computer code

Description automatically generated

**Kết quả:**

**A screenshot of a graph

Description automatically generated**

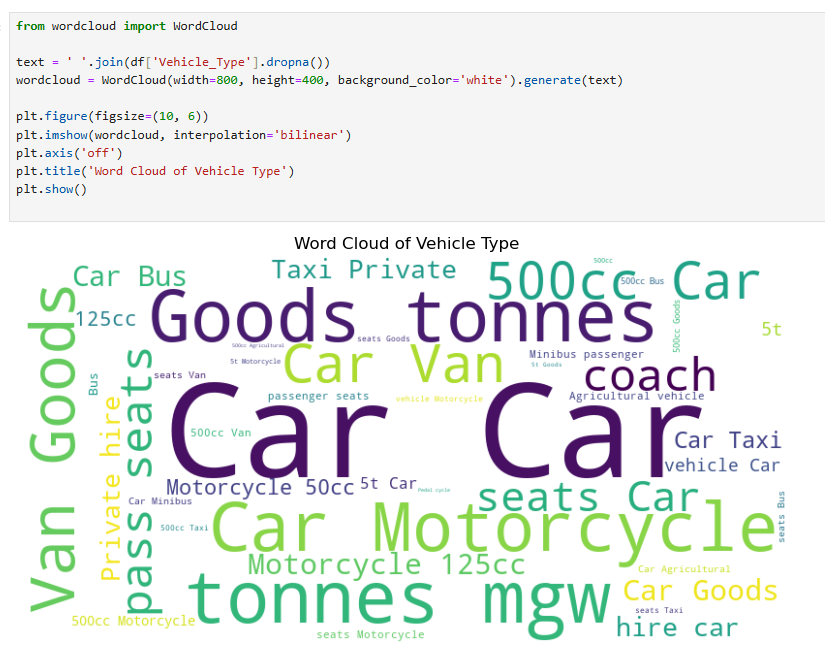
### Trực quan hóa dữ liệu

* **Tiến hành thống kê mức độ nghiêm trọng của các vụ tai nạn:**

A graph with blue and orange bars

Description automatically generated

* Từ biểu đồ trên ta thấy số vụ tai nạn ở mức độ “Slight” là rất cao, hơn 250000 vụ, chiếm đại đa số, và gấp khoảng 5000 lần số vụ tại nạn có mức độ “Fetal” là 49 vụ.
* **Các loại phương tiện gặp tai nạn phổ biến nhất:**



* Kích thước càng lớn tương ứng với độ phổ biến càng cao, hình minh họa cho thấy “Car” là loại phương tiện gặp tai nạn nhiều nhất.
* **Thống kê số vụ tai nạn theo các ngày trong tuần:**

A graph of accident statistics

Description automatically generated with medium confidence

* Từ biểu đồ ta thấy “Friday” là ngày có số vụ tai nạn xảy ra nhiều nhất, và “Sunday” số lượng vụ tai nạn xảy ra ít nhất.
* **Số vụ tai nạn theo điều kiện thời tiết:**

A graph with blue squares

Description automatically generated

* Loại thời tiết “Fine no high winds” xảy ra nhiều vụ tai nạn nhất, cao hơn hẳn số vụ tai nạn có thời tiết là “Raining no high winds”.

## Mô tả bài toán

Đồ án của chúng tôi sẽ dự đoán loại tai nạn dựa trên việc phân tích các đặc điểm được ghi lại trong dataset. Mục tiêu của đồ án là cung cấp một cái nhìn sâu sắc về những yếu tố làm tăng mức độ nghiêm trọng của tai nạn, và xác định các khu vực hoặc tình huống có nguy cơ cao tiềm tàng.

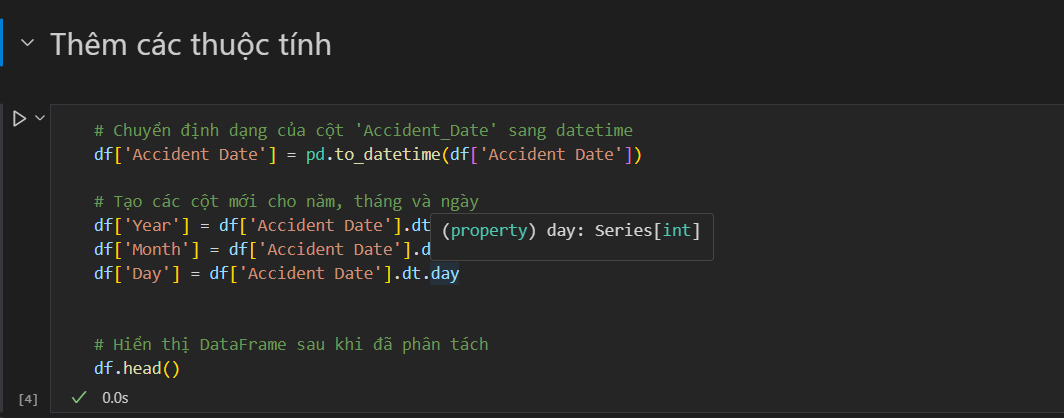
## Công cụ thực hiện

* Python: version 3.11
* Jupyter notebook
* Visual studio Code
* Các thư viện trong Python

# XỬ LÝ TIỀN DỮ LIỆU

## Thêm các thuộc tính

Phân tách thuộc tính Accident Date theo năm, tháng, ngày để cung cấp cho các mô hình.

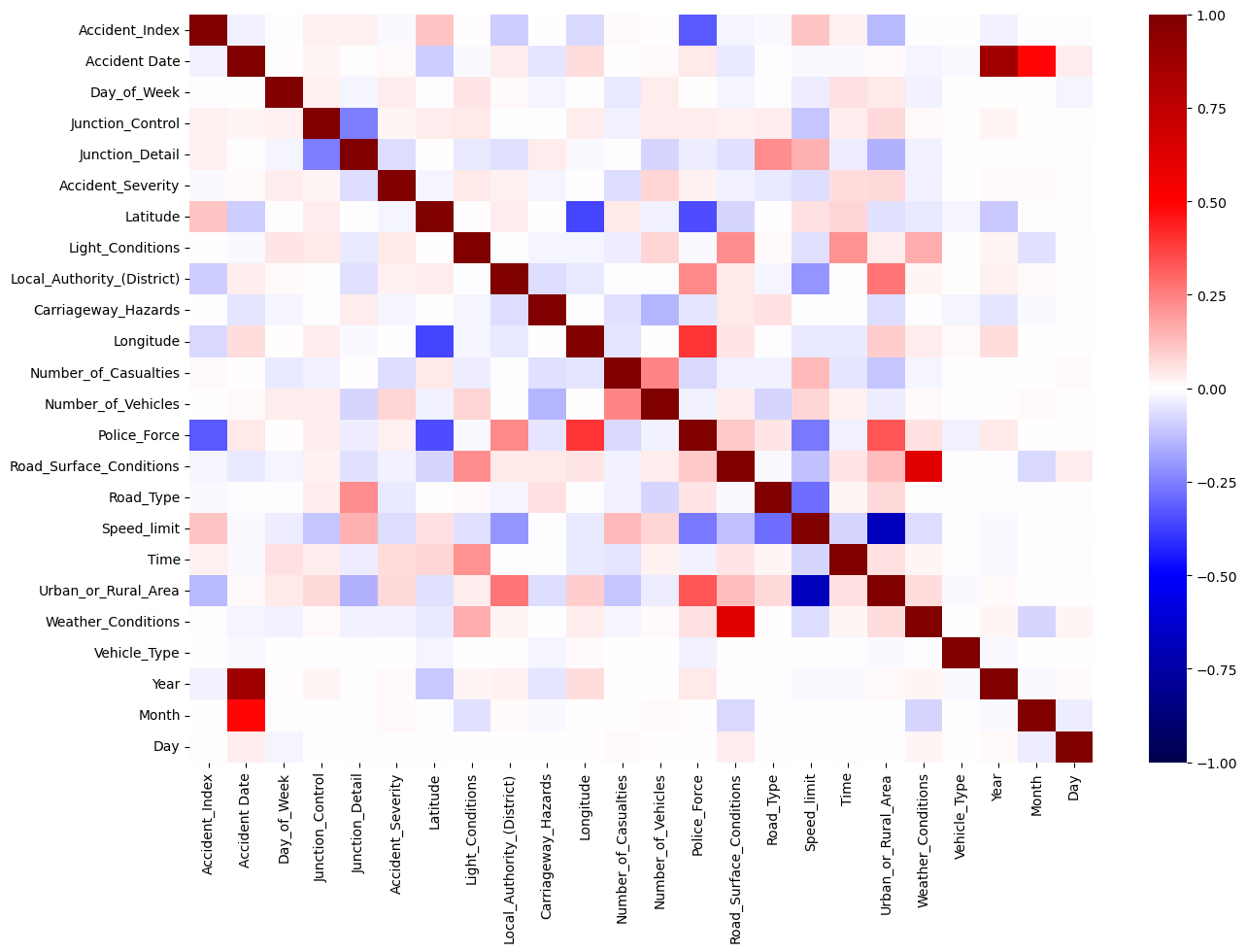




## Kiểm tra độ tương quan giữa các thuộc tính

Trong khối tiếp theo, ma trận tương quan giữa tất cả các thuộc tính có thể được trình bày dưới dạng bản đồ nhiệt. Cùng với đó, có thể quan sát mối tương quan giữa các thuộc tính khác nhau của dữ liệu, để kiểm tra xem một số tính năng có tương quan cao hay không và loại bỏ một số trong chúng.





Từ Heatmap ta có nhận xét sau:

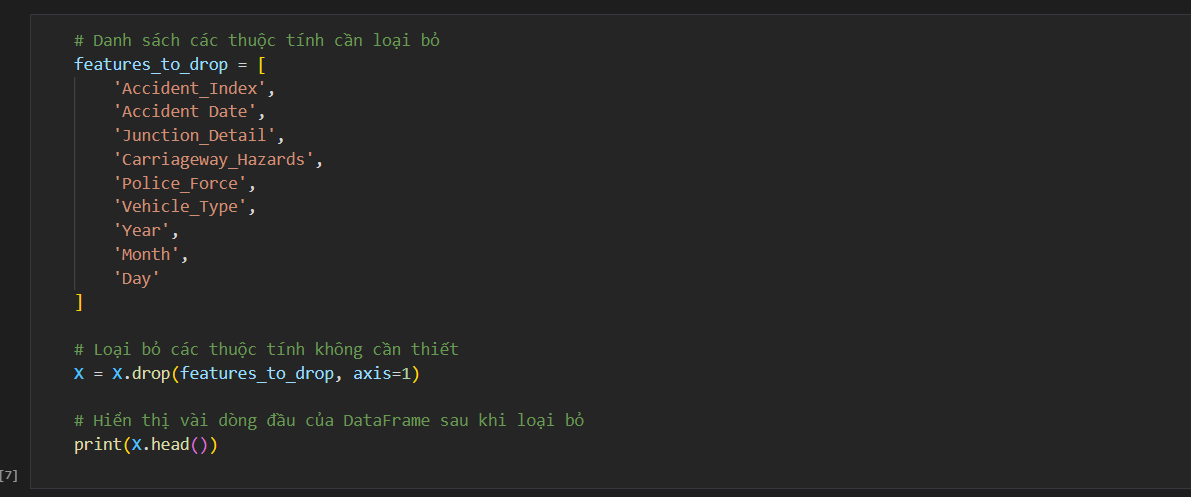
* Accident\_Index: Không cần thiết vì đây là chỉ số định danh duy nhất.
* Accident Date: không có mối tương quan mạnh với các thuộc tính khác.
* Day\_of\_Week: Có tương quan.
* Junction\_Control: Tương quan cao với "Junction\_Detail".
* Junction\_Detail: Tương quan cao với "Junction\_Control".
* Accident\_Severity: Tương quan cao với "Number\_of\_Casualties".
* Latitude: Tương quan cao với "Longitude".
* Light\_Conditions: Có một số tương quan
* Local\_Authority\_(District): Tương quan cao với "Police\_Force".
* Carriageway\_Hazards: Không rõ ràng.
* Longitude: Tương quan cao với "Latitude".
* Number\_of\_Casualties: Tương quan cao với "Accident\_Severity" và "Number\_of\_Vehicles".
* Number\_of\_Vehicles: Tương quan cao với "Number\_of\_Casualties".
* Police\_Force: Tương quan cao với "Local\_Authority\_(District)".
* Road\_Surface\_Conditions: Có tương quan.
* Road\_Type: Tương quan cao với "Speed\_limit" và "Urban\_or\_Rural\_Area".
* Speed\_limit: Tương quan cao với "Road\_Type".
* Time: Có tương quan.
* Urb n\_or\_Rural\_Area: Tương quan cao với "Road\_Type".
* Weather\_Conditions: Có tương quan.
* Vehicle\_Type: Không rõ ràng.
* Year: Tương quan thấp.
* Month: Tương quan thấp.
* Day: Tương quan thấp.

Police\_Force (Tương quan cao với "Local\_Authority\_(District)" - trùng lặp)

Junction\_Detail (Tương quan cao với "Junction\_Control" - trùng lặp)

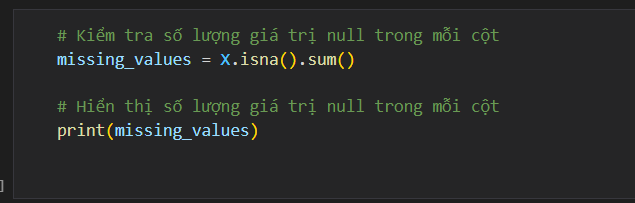
## Lựa chọn thuộc tính

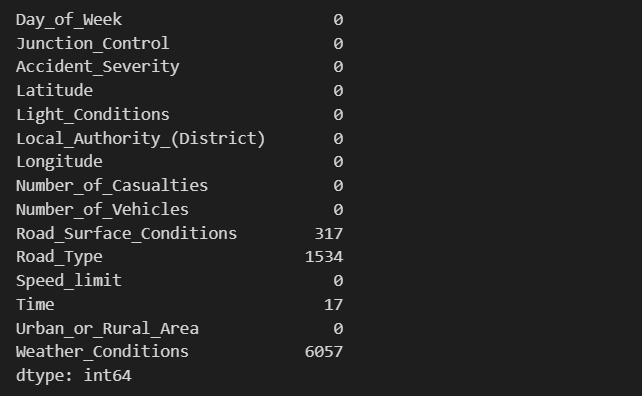
Chọn các thuộc tính có độ tương quan cao, bỏ các thuộc tính trùng lặp và độ tương quan thấp.



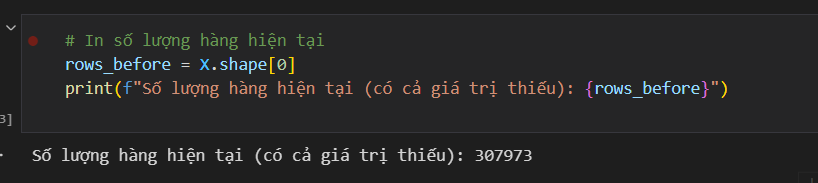
## Kiểm tra dữ liệu null

Kiểm tra số lượng giá trị null trong mỗi cột

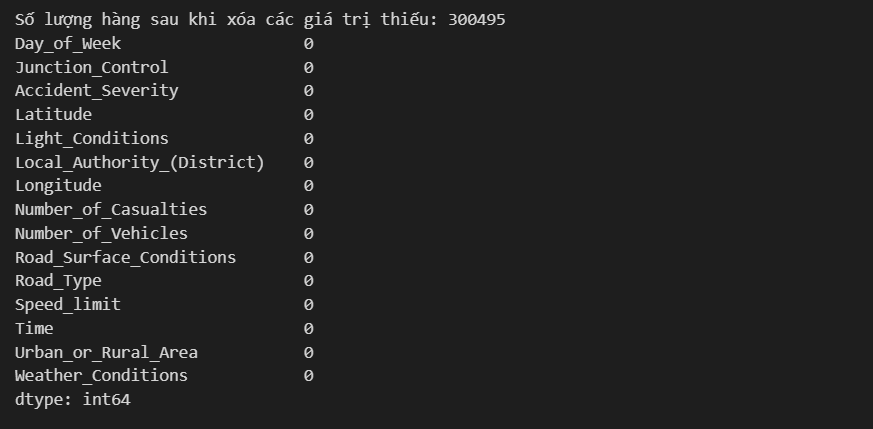
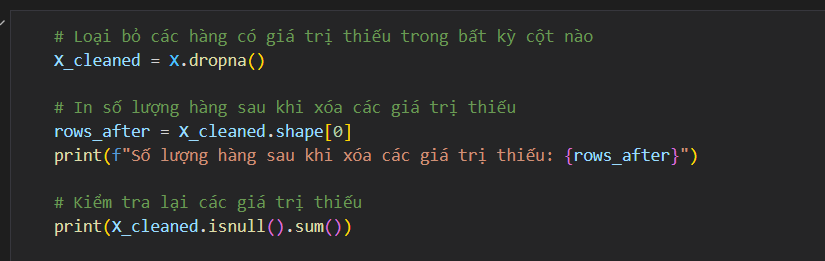




* Số lượng dòng hiện tại

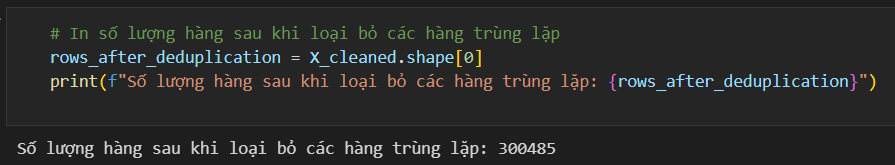
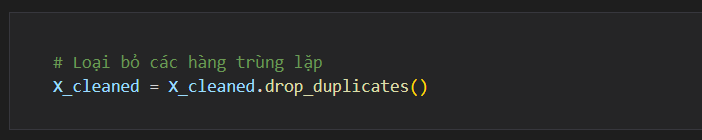


Ta thấy số lượng dữ liệu bị thiếu không đáng kể so với lượng dữ liệu hiện có trong dataset, nên ta sẽ xóa các hàng chứa dữ liệu bị thiếu.



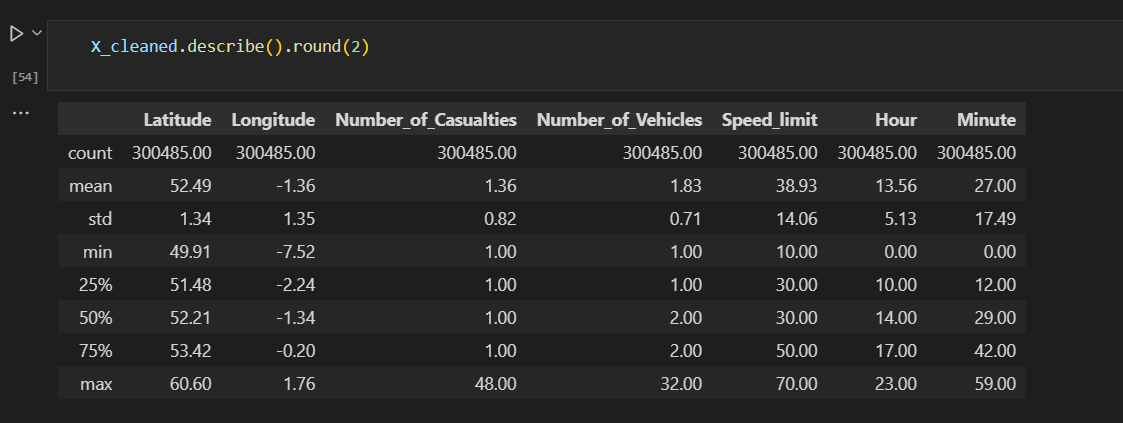
## Loại bỏ trùng lặp

Trong phần này, chúng ta sẽ kiểm tra xem có một số bản sao trong tập dữ liệu hay không.



## Kiểm tra phương sai của các thuộc tính

Trong phần này, chúng ta sẽ kiểm tra phương sai của từng tính năng để loại bỏ các tính năng có phương sai rất thấp vì chúng không thể giúp phân biệt các trường hợp.



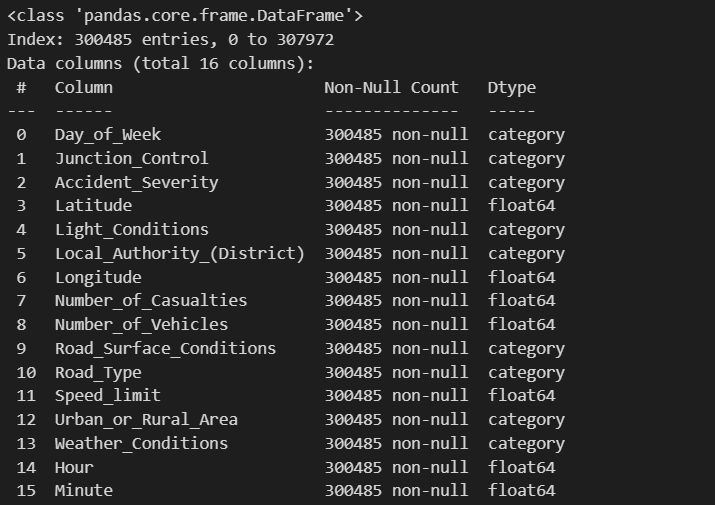
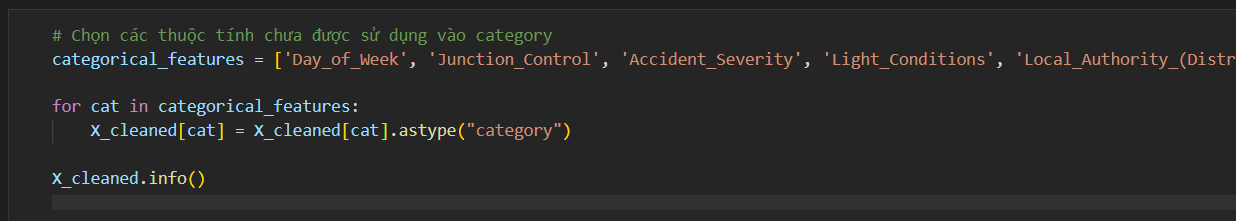
1. Chia tỷ lệ thuộc tính

Trong phần này, nhóm chia tỷ lệ và chuẩn hóa các thuộc tính. Để cải thiện hiệu suất của các mô hình, nhóm đã chuẩn hóa các giá trị của các thuộc tính liên tục.

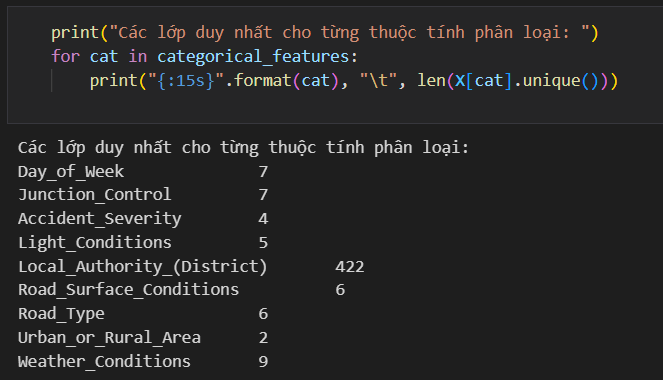


1. **Mã hóa tính năng**

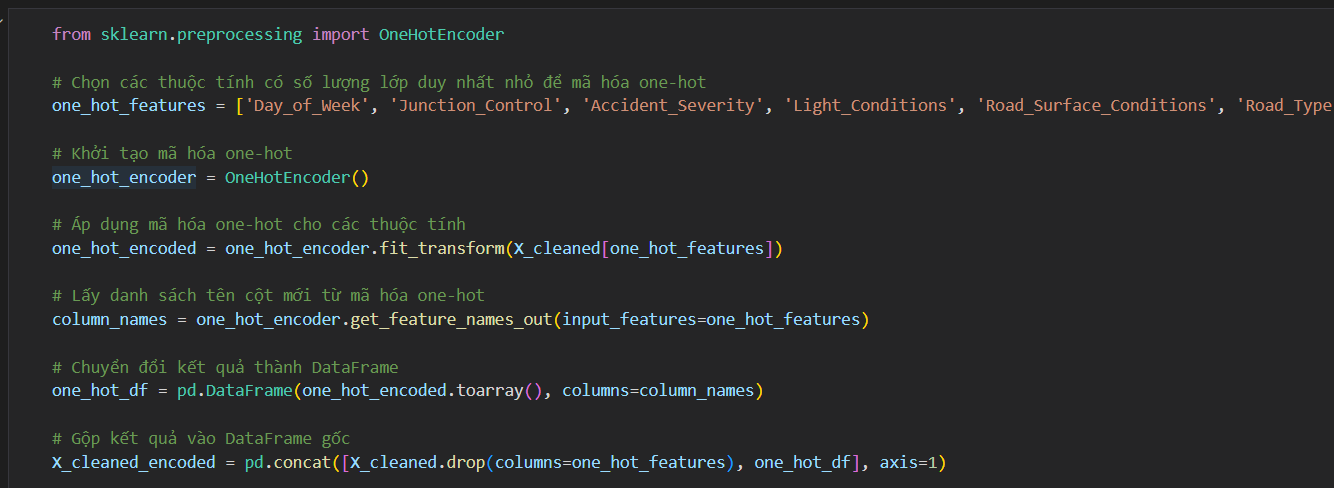
Trong phần này, nhóm sẽ mã hóa các thuộc tính phân loại.



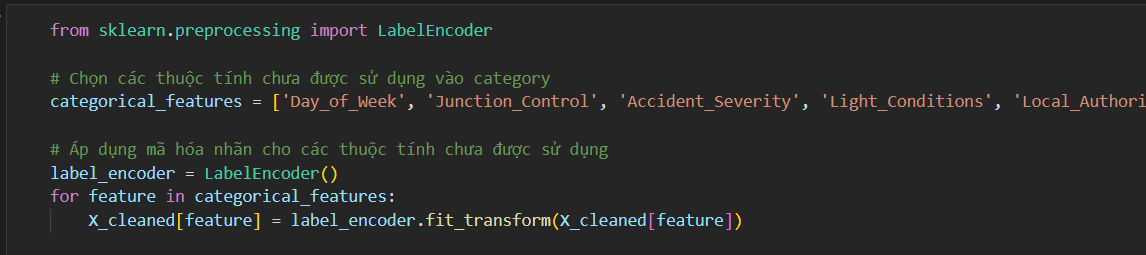
Số lượng lớp duy nhất cho từng tính năng phân loại.

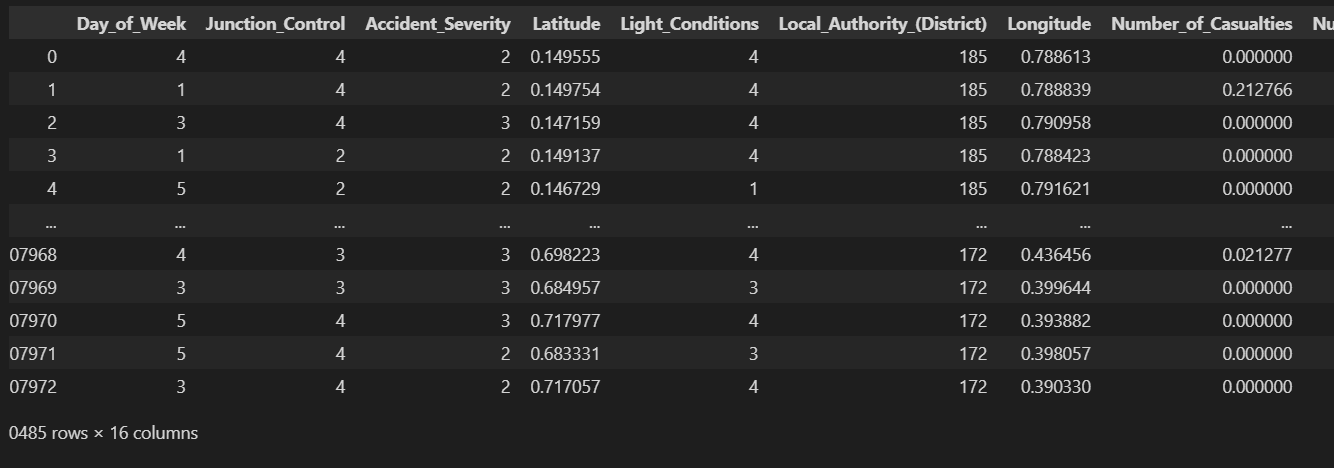


Sử dụng one-hot encoding cho các thuộc tính có số lượng lớp duy nhất nhỏ (thường ít hơn khoảng vài chục).



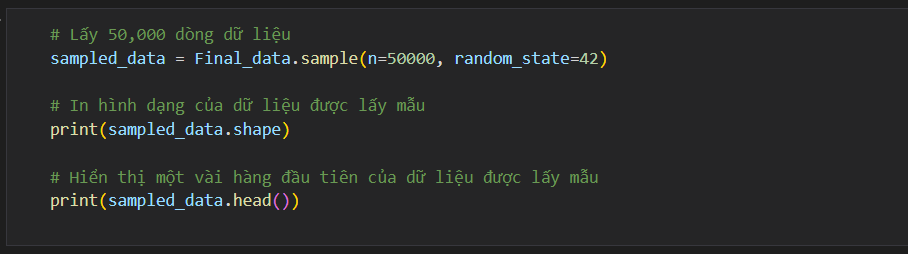
Sử dụng label encoding cho các thuộc tính có số lượng lớp duy nhất lớn (ví dụ: Local\_Authority\_(District))





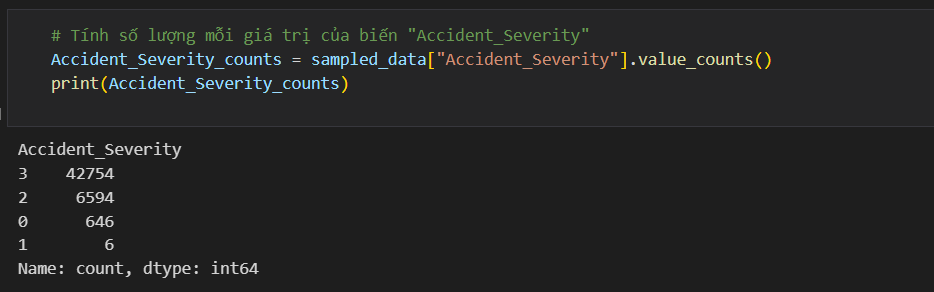
1. Cân bằng dữ liệu

* Lấy 50000 dòng dữ liệu ngẫu nhiên từ Final\_Data để tiến hành cân bằng dữ liệu, tránh việc tạo ra quá nhiều dòng dữ liệu khi chạy sẽ tốn tài nguyên và thời gian.

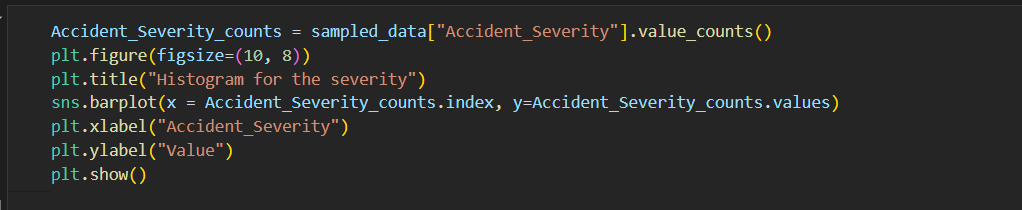


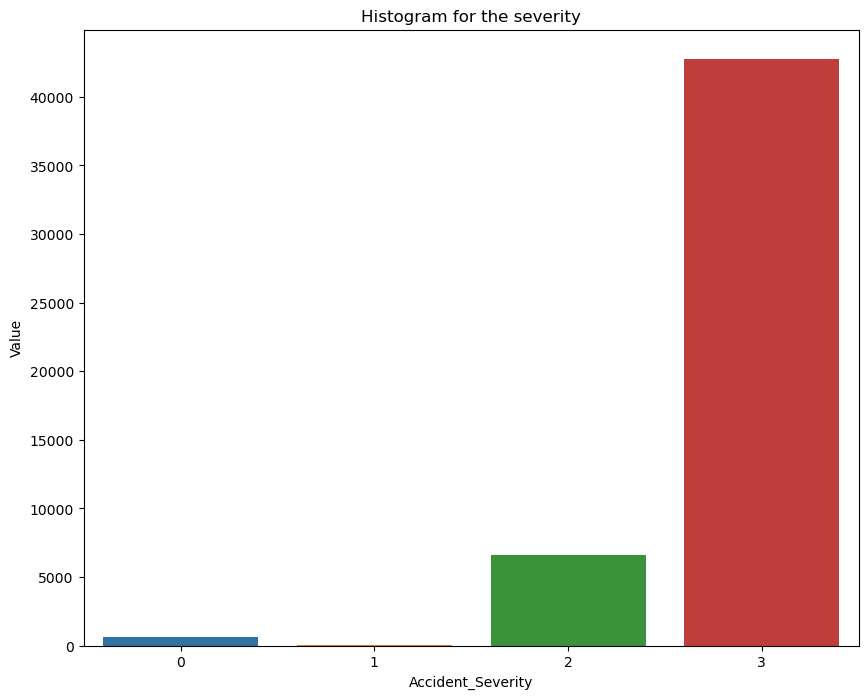
* Tiến hành cân bằng dữ liệu

Đầu tiên, ta tính số lượng mỗi giá trị của biến "Accident\_Severity".



* Hiển thị biểu đồ thanh cho mức độ nghiêm trọng của các vụ tai nạn

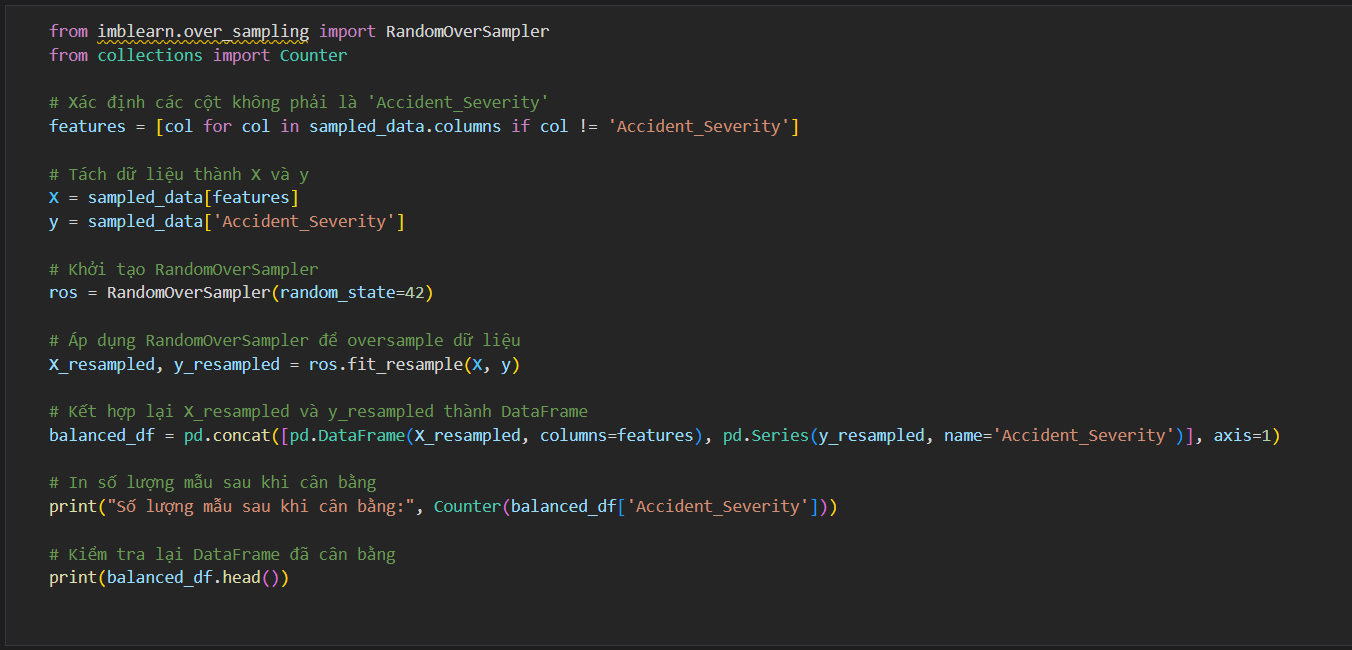




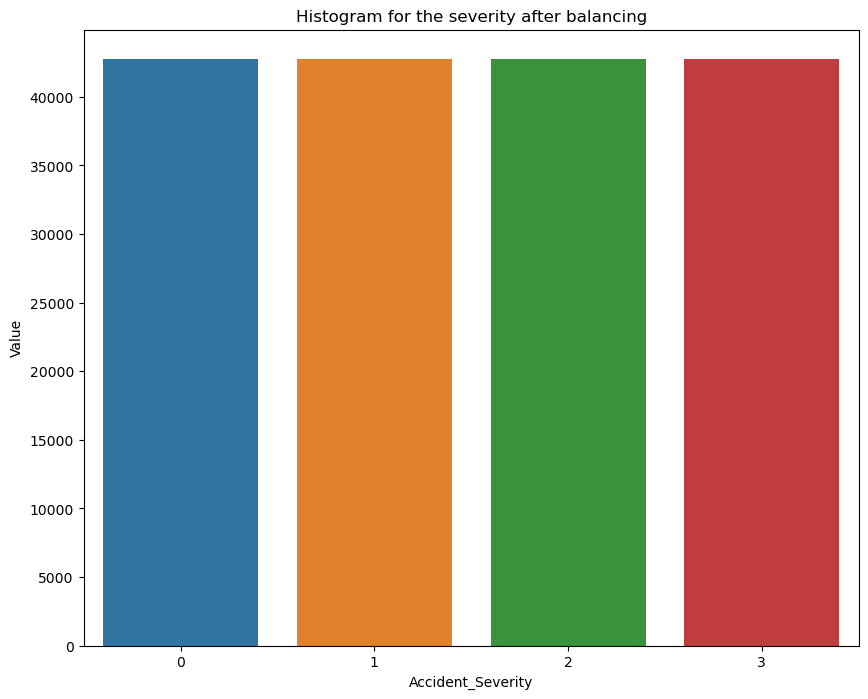
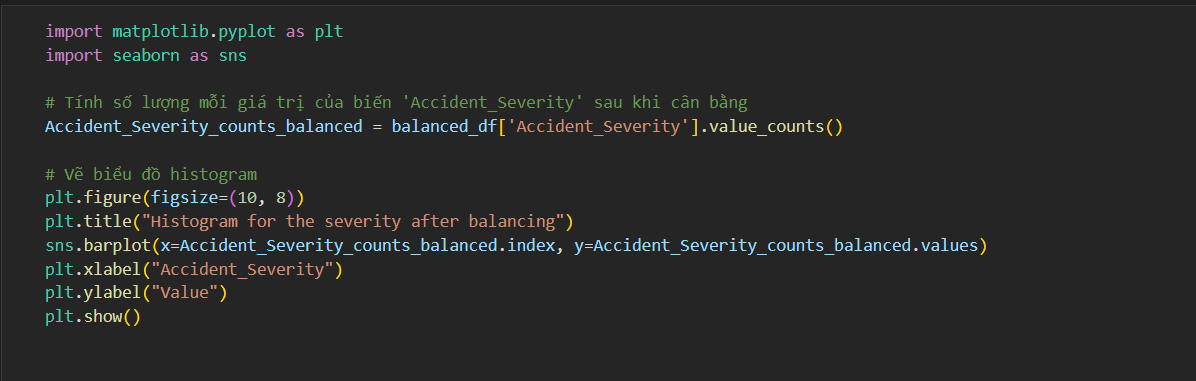
Dựa vào biểu đồ ta thấy có sự mất cân bằng, dữ liệu của Accident\_Severity = 1 rất thấp so với Accident\_Severity = 3

Oversampling là một kỹ thuật được sử dụng để xử lý sự mất cân bằng dữ liệu, khi mà số lượng mẫu trong các lớp không đều nhau. Trong trường hợp mất cân bằng dữ liệu, một số lớp có rất nhiều mẫu, trong khi các lớp khác lại có rất ít mẫu. Điều này có thể gây ra vấn đề cho các thuật toán học máy, vì chúng có thể học lệch về các lớp có nhiều mẫu hơn.

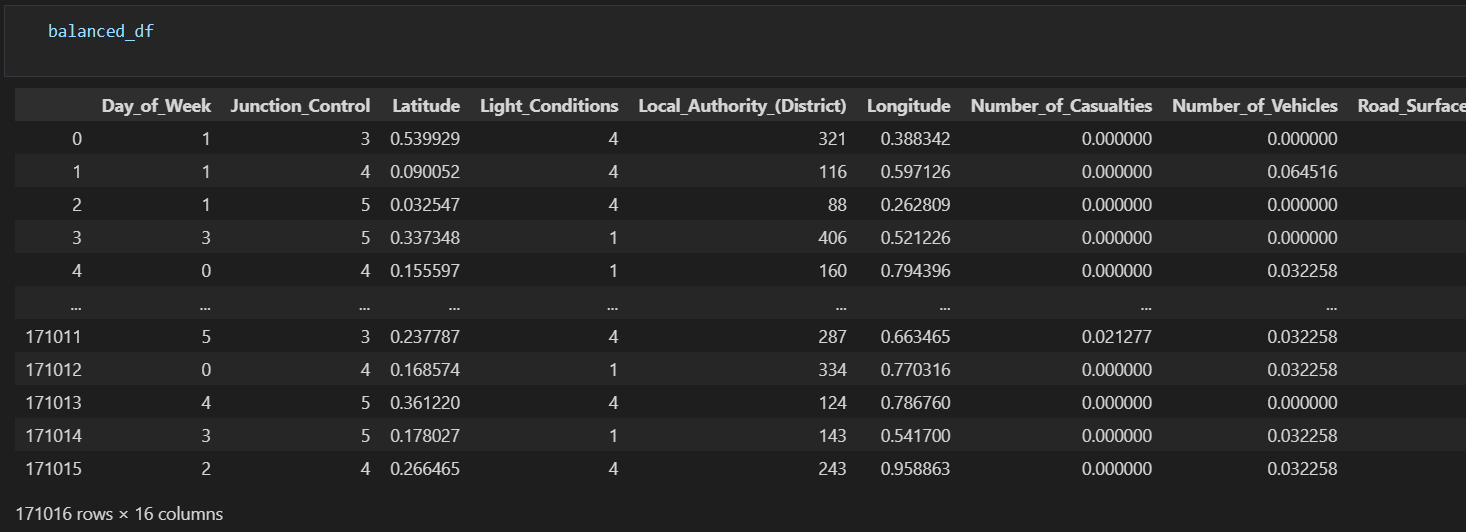
Random Oversampling: Tạo thêm các bản sao của các mẫu trong lớp thiểu số một cách ngẫu nhiên cho đến khi các lớp được cân bằng. Phương pháp này đơn giản nhưng có thể dẫn đến overfitting vì nó lặp lại các mẫu giống nhau.

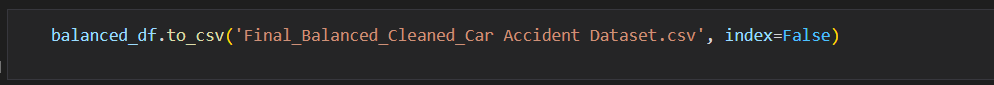


* Hiển thị biểu đồ thanh cho mức độ nghiêm trọng của các vụ tai nạn sau khi cân bằng dữ liệu.



* Dữ liệu sau khi được cân bằng





Xuất file csv, đây cũng là data cuối cùng đã được làm sạch và cân bằng, dùng cho các phần tiếp theo.

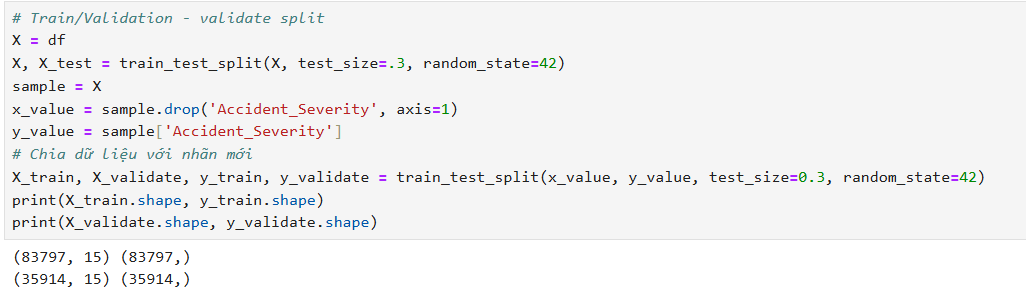
# ÁP DỤNG CÁC THUẬT TOÁN KHAI THÁC

## Chia dữ liệu

Sau khi tiền xử lý dữ liệu, chúng tôi đã thu được DataFrame df rồi gắn nó vào X cuối cùng với các tính năng phân loại được mã hóa.

Chia DataFrame thành 2 phần: train(x) và test(x\_test) với tỉ lệ 70% train và 30% test cho dự đoán.

Sau đó lại chia nhỏ train thành hai phần với nhãn mới để huấn luyện mô hình gồm pha train và pha validate.  
Với random\_state = 42 để đảm bảo rằng việc chia dữ liệu sẽ luôn cho kết quả giống nhau mỗi khi mã được chạy lại.



## KNN

### Giới thiệu thuật toán

Thuật toán K-Nearest Neighbors (KNN) là một phương pháp học máy có giám sát được sử dụng để giải quyết các vấn đề phân loại và hồi quy. KNN là một trong những thuật toán phân loại cơ bản nhưng cần thiết nhất trong học máy. Nó thuộc miền học có giám sát và có ứng dụng mạnh mẽ trong nhận dạng mẫu, khai thác dữ liệu và phát hiện xâm nhập.

Thuật toán K-NN hoạt động bằng cách tìm K lân cận gần nhất với một điểm dữ liệu nhất định dựa trên thước đo khoảng cách, chẳng hạn như khoảng cách Euclide, Manhattan, Minkowski,... Lớp hoặc giá trị của điểm dữ liệu sau đó được xác định bằng đa số phiếu bầu hoặc mức trung bình của K hàng xóm. Cách tiếp cận này cho phép thuật toán thích ứng với các mẫu khác nhau và đưa ra dự đoán dựa trên cấu trúc cục bộ của dữ liệu.

Thuật toán KNN có 4 bước như sau:

* Bước 1: Chọn giá trị tối ưu của K

K đại diện cho số lượng hàng xóm gần nhất cần được xem xét khi đưa ra dự đoán.

* Bước 2: Tính khoảng cách

Để đo lường sự tương tự giữa điểm dữ liệu mục tiêu và điểm dữ liệu huấn luyện, khoảng cách Euclide được sử dụng. Khoảng cách được tính giữa từng điểm dữ liệu trong tập dữ liệu và điểm mục tiêu.

* Bước 3: Tìm hàng xóm gần nhất

K điểm dữ liệu có khoảng cách nhỏ nhất tới điểm đích là những điểm lân cận gần nhất.

* Bước 4: Bỏ phiếu phân loại hoặc lấy trung bình để hồi quy

Trong bài toán phân loại, nhãn lớp được xác định bằng cách biểu quyết theo đa số. Lớp có nhiều lần xuất hiện nhất trong số các lớp lân cận sẽ trở thành lớp được dự đoán cho điểm dữ liệu mục tiêu.

Trong bài toán hồi quy, nhãn lớp được tính bằng cách lấy trung bình các giá trị đích của K lân cận gần nhất. Giá trị trung bình được tính toán sẽ trở thành đầu ra dự đoán cho điểm dữ liệu mục tiêu.

### Lý do chọn thuật toán

* KNN có thể xử lý dữ liệu đa dạng: Tập dữ liệu của nhóm bao gồm dữ liệu số và dữ liệu phân loại (“Day\_of\_Week”, “Junction\_Control”, “Local\_Authority”, “Accident\_Severity”,..)
* KNN tương đối đơn giản và dễ hiểu, hoạt động dựa trên nguyên tắc cơ bản của khoảng cách giữa các điểm dữ liệu.
* Thích hợp cho các bài toán phân loại (“Accident\_Severity”)

### Xây dựng và dự đoán với mô hình

* **Bước 1:** Sử dụng ‘StandardScaler” từ thư viện ‘sklearn.preprocessing’ để chuẩn hóa các đặc trưng
* ‘with\_mean=False’ được sử dụng để tránh việc trừ đi trung bình (centering) của dữ liệu. Vì điều này sẽ làm mất tính thưa thớt của ma trận.

**Ảnh có chứa văn bản, Phông chữ, số, hàng

Mô tả được tạo tự động**

* **Bước 2:** Khởi tạo và huấn luyện KNN
* Sử dụng ‘KNeighborsClassifier’ từ thư viện sklearn.neighbors
* Trong trường hợp này, không có tham số nào được truyền vào KNeighborsClassifier(), vì vậy nó sẽ sử dụng giá trị mặc định của n\_neighbors=5, nghĩa là sử dụng 5 láng giềng gần nhất để phân loại.

**Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ, hàng

Mô tả được tạo tự động**

* **Bước 5:** Dự đoán và thể hiện kết quả trên ma trận nhầm lẫn
* **‘predict’** được sử dụng để dự đoán nhãn của các mẫu trong tập kiểm tra (X\_test).
* **‘ConfusionMatrixDisplay.from\_estimator’** từ thư viện sklearn.metrics được sử dụng để hiển thị ma trận nhầm lẫn (confusion matrix) dựa trên dự đoán của mô hình trên tập kiểm tra (X\_test) và nhãn thực tế (y\_test).

Ảnh có chứa văn bản, ảnh chụp màn hình, biểu đồ, hình vuông

Mô tả được tạo tự động

* **Bước 6:** Tạo biểu đồ cột để đại diện cho độ quan trọng của tính năng, được xác định bởi thuật toán KNN. Độ quan trọng của một tính năng càng cao thì tính năng đó càng ảnh hưởng đến việc dự đoán với mô hình KNN.

Ảnh có chứa văn bản, ảnh chụp màn hình, Song song, Phông chữ

Mô tả được tạo tự động

## Decision tree

### Giới thiệu thuật toán Decision tree

Decision Tree là một trong những thuật toán học máy phổ biến nhất, đặc biệt trong các bài toán phân loại và hồi quy. Thuật toán này hoạt động bằng cách xây dựng một mô hình cây quyết định dựa trên các giá trị của các thuộc tính đầu vào để đưa ra quyết định hoặc dự đoán.

Cây quyết định là một cấu trúc dữ liệu có dạng cây, trong đó:

* Nút gốc (root node): Nút bắt đầu của cây.
* Nút lá (leaf node): Các nút cuối cùng không có nút con, đại diện cho các quyết định hoặc phân loại cuối cùng.
* Nút trung gian (internal node): Các nút có nút con, đại diện cho các thuộc tính được sử dụng để chia dữ liệu tại mỗi bước..

Trong cây quyết định, mỗi nút nội bộ chia không gian mẫu thành hai hoặc nhiều không gian con theo một hàm rời rạc nhất định của các giá trị thuộc tính đầu vào. Trong trường hợp đơn giản và phổ biến nhất, mỗi kiểm tra xem xét một thuộc tính đơn lẻ, sao cho không gian mẫu được chia theo giá trị của thuộc tính đó. Trong trường hợp các thuộc tính số, điều kiện đề cập đến một khoảng.

Mỗi lá được gán cho một lớp đại diện cho giá trị mục tiêu phù hợp nhất. Ngoài ra, lá có thể giữ một vector xác suất chỉ ra xác suất của thuộc tính mục tiêu có một giá trị nhất định. Các mẫu được phân loại bằng cách điều hướng chúng từ gốc của cây xuống một lá, theo kết quả của các kiểm tra dọc theo đường đi.

### Lý do chọn thuật toán

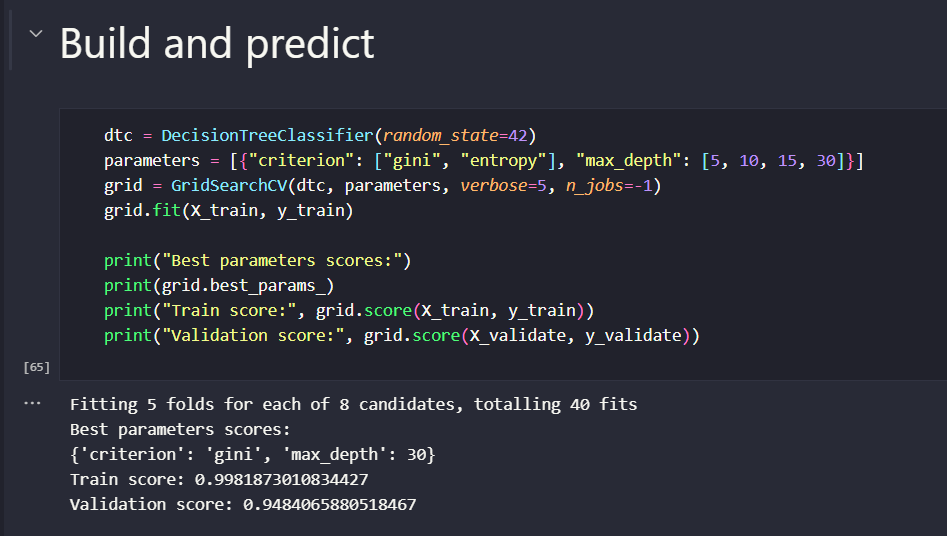
Một số ưu điểm của cây quyết định là:

* Đơn giản để hiểu và diễn giải. Cây có thể được trực quan hóa.
* Yêu cầu chuẩn bị dữ liệu ít. Các kỹ thuật khác thường yêu cầu chuẩn hóa dữ liệu, tạo các biến giả và loại bỏ các giá trị trống. Tuy nhiên, lưu ý rằng module này không hỗ trợ các giá trị bị thiếu.
* Chi phí sử dụng cây (tức là dự đoán dữ liệu) là logarithmic theo số lượng điểm dữ liệu được sử dụng để huấn luyện cây.
* Có thể xử lý cả dữ liệu số và dữ liệu phân loại.
* Có thể xử lý các vấn đề đa đầu ra.
* Sử dụng mô hình hộp trắng. Nếu một tình huống nhất định có thể quan sát trong mô hình, lời giải thích cho điều kiện đó dễ dàng được diễn giải bằng logic boolean. Ngược lại, trong mô hình hộp đen (ví dụ, trong mạng nơ-ron nhân tạo), kết quả có thể khó giải thích hơn.
* Có thể xác thực mô hình bằng các bài kiểm tra thống kê. Điều đó làm cho có thể đánh giá độ tin cậy của mô hình.
* Hiệu suất tốt ngay cả khi các giả định của nó bị vi phạm một phần bởi mô hình thực từ đó dữ liệu được tạo ra.

### Xây dựng và dự đoán với mô hình

**Bước 1:**

* Tạo một mô hình cây quyết định (dtc) sử dụng DecisionTreeClassifier từ sklearn.tree, với tham số random\_state được đặt là 42.
* Tạo một danh sách các tham số để thử nghiệm trong một grid search. Danh sách này bao gồm hai tham số: criterion (phương pháp đo độ tinh khiết của nút) và max\_depth (độ sâu tối đa của cây).
* Tạo một đối tượng grid search với mô hình cây quyết định ‘dtc’, các tham số để thử nghiệm ‘parameters’, ‘verbose=5’ để in kết quả của grid search, và n\_jobs=-1 để sử dụng tất cả các lõi CPU có sẵn.
* Huấn luyện mô hình trên tập huấn luyện ‘X\_train’ và ‘y\_train’, và thực hiện grid search để tìm các tham số tốt nhất cho mô hình. Sau đó, in ra các tham số tốt nhất, điểm số trên tập huấn luyện và điểm số trên tập xác thực.



* Đầu ra có nghĩa là mô hình Cây Quyết Định đã được huấn luyện và các tham số tốt nhất là “criterion: gini” và “max\_depth: 10”.
* Điểm số độ chính xác trên dữ liệu huấn luyện xấp xỉ 0.998, cho thấy mô hình đã dự đoán chính xác nhãn lớp cho 99.8% ví dụ trong tập huấn luyện.
* Điểm số độ chính xác trên dữ liệu xác thực xấp xỉ 0.9484, cho thấy mô hình đã dự đoán chính xác nhãn lớp cho 94.84% ví dụ trong tập xác thực.

**Bước 2:** Huấn luyện mô hình Cây Quyết Định (dtc) trên tập dữ liệu huấn luyện (X\_train và y\_train) và in ra điểm số trên tập huấn luyện và tập xác thực với các siêu tham số mặc định.

**A screenshot of a computer

Description automatically generated**

* Điểm số trên tập huấn luyện xấp xỉ 0.999 cho thấy mô hình đã dự đoán chính xác 99.9% các mẫu trong tập huấn luyện.
* Điểm số trên tập xác thực xấp xỉ 0.949 cho thấy mô hình đã dự đoán chính xác 94.9% các mẫu trong tập xác thực.

**Bước 3:** Tạo một DataFrame từ kết quả của grid search và sắp xếp kết quả theo thứ tự tăng dần của điểm số trên tập xác thực.

A screenshot of a computer

Description automatically generated

**Bước 4:**

* Dự đoán nhãn của tập dữ liệu xác thực (X\_validate) bằng mô hình cây quyết định (dtc) đã huấn luyện.
* Tính ma trận nhầm lẫn bằng cách so sánh các nhãn thực sự (y\_validate) với các nhãn dự đoán (y\_pred).
* Tạo một ma trận nhầm lẫn để trực quan hóa hiệu suất của mô hình. Ma trận nhầm lẫn được lưu trữ trong một DataFrame của Pandas và hiển thị bằng cách sử dụng heatmap từ thư viện Seaborn.

A screenshot of a computer

Description automatically generated

**Bước 5:**

* Tìm ra 30 đặc trưng quan trọng nhất bằng cách tạo một DataFrame với số hàng bằng số cột trong X\_train và một cột tên là “importance”, sử dụng tên các đặc trưng trong X\_train làm chỉ mục.

A screen shot of a computer

Description automatically generated

**Bước 6:**

* Tạo một biểu đồ của mô hình Cây Quyết Định 'dtc' bằng cách sử dụng hàm plot\_tree() từ sklearn.tree, với max\_depth = 4.

A screenshot of a computer

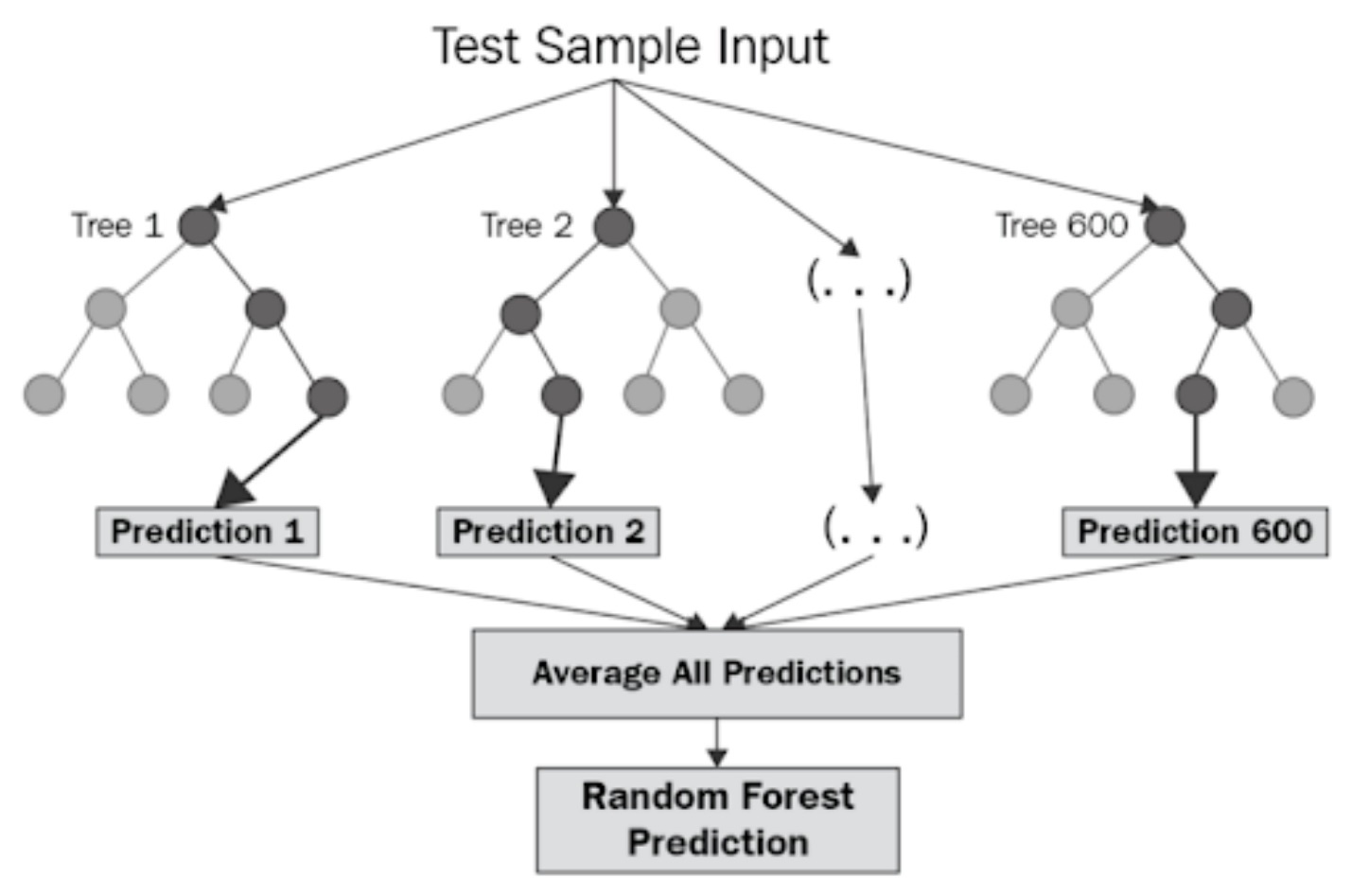
Description automatically generated

## Random forest

### Giới thiệu thuật toán Random Forest (RF)

Thuật toán Random Forest là một kỹ thuật học máy phổ biến được sử dụng cho cả các nhiệm vụ phân loại và hồi quy. Đây là phương pháp học tập tổng hợp, kết hợp các dự đoán của nhiều cây quyết định để tạo ra dự đoán cuối cùng. Random Forest được biết đến với độ mạnh và độ chính xác cao, khiến chúng được sử dụng rộng rãi trong các lĩnh vực khác nhau, bao gồm tài chính, chăm sóc sức khỏe và xử lý ngôn ngữ tự nhiên.

Hình minh họa cấu trúc rừng ngẫu nhiên:



### Lý do chọn thuật toán:

Một trong những ưu điểm của Random Forest là khả năng xử lý các tập dữ liệu lớn với nhiều chiều. Chúng cũng ít bị quá khớp (overfitting) hơn so với các cây quyết định riêng lẻ. Ngoài ra, Random Forest có thể cung cấp các phép đo về tầm quan trọng của đặc trưng, cho phép xác định các đặc trưng có ảnh hưởng lớn nhất trong tập dữ liệu.

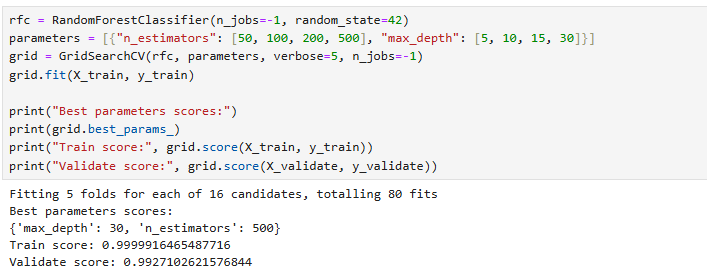
Tuy nhiên, cần lưu ý rằng Random Forests có thể tốn nhiều thời gian tính toán, đặc biệt là khi xử lý một số lượng lớn cây hoặc tập dữ liệu phức tạp. Ngoài ra, chúng có thể không hoạt động tốt trên các tập dữ liệu có mối quan hệ tuyến tính mạnh, vì cây quyết định không phải lúc nào cũng phù hợp để nắm bắt các mẫu mẫu này.

* Tổng thể, thuật toán Random Forest là một kỹ thuật mạnh mẽ và linh hoạt cho các nhiệm vụ học máy, cung cấp một sự cân bằng tốt giữa độ chính xác, khả năng giải thích và tính ổn định.

### Xây dựng và dự đoán với mô hình RF

**Bước 1:**

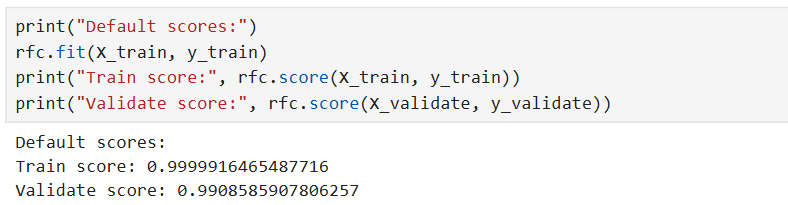
* Thực hiện bộ phân loại Random Forest: Đoạn mã khởi tạo một bộ phân loại
* Random Forest (rfc) bằng cách sử dụng lớp Random Forest Classifier.
* Đối tượng bộ phân loại Random Forest được khởi tạo với n\_jobs và
* random\_state.
* Các siêu tham số cần được điều chỉnh được chỉ định trong một từ điển được
* gọi là parameters.
* Một đối tượng GridSearchCV được tạo ra và phù hợp với dữ liệu huấn
* luyện.
* Các siêu tham số tốt nhất được xuất ra cùng với các điểm số độ chính xác của mô hình trên dữ liệu huấn luyện và dữ liệu xác thực.



* Kết quả đầu ra cho thấy rằng mô hình bộ phân loại Random Forest đã được huấn luyện và đánh giá bằng cách sử dụng GridSearchCV với 5-fold cross-validation trên 2 tổ hợp siêu tham số (n\_estimators, max\_depth).
* Điểm số độ chính xác trên dữ liệu huấn luyện là 0,999, cho thấy rằng mô hình đã dự đoán đúng nhãn lớp cho 99,9% các ví dụ trong tập huấn luyện.
* Điểm số độ chính xác trên dữ liệu xác thực là 0,9927, cho thấy rằng mô hình đã dự đoán đúng nhãn lớp cho 99,27% các ví dụ trong tập xác thực.

**Bước 2:**

* Sử dụng mô hình bộ phân loại Random Forest (rfc) để phù hợp với tập dữ liệu huấn luyện (X\_train và y\_train) và in ra các điểm số huấn luyện và xác thực với các siêu tham số mặc định.



* Điểm số huấn luyện là 0,999 cho thấy rằng mô hình đã dự đoán đúng 99.9 % các mẫu huấn luyện.
* Điểm số xác thực là 0,9908 cho thấy rằng mô hình đã dự đoán đúng 99.08% các mẫu xác thực.

**Bước 3:**

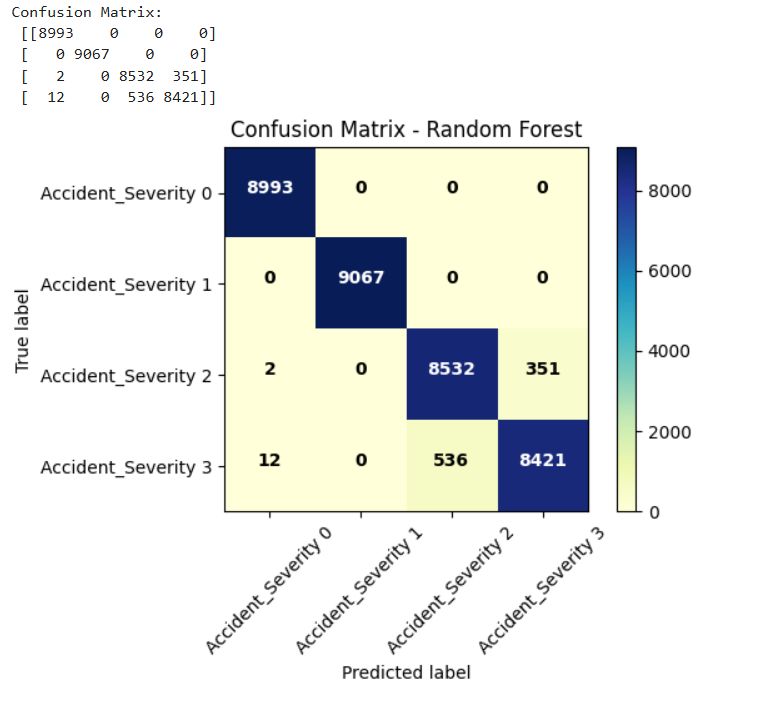
* Tạo một DataFrame Pandas chứa kết quả của cross-validation lưới tìm kiếm và sắp xếp theo rank\_test\_score.

A screenshot of a computer

Description automatically generated

Kết quả DataFrame chứa một hàng cho mỗi tổ hợp siêu tham số và các cột cho các số liệu đánh giá khác nhau, bao gồm điểm số trung bình trên tập kiểm tra, độ lệch chuẩn của điểm số kiểm tra và các giá trị của siêu tham số.

**Bước 4:** Tạo ma trận nhầm lẫn để trực quan hóa hiệu suất của mô hình.



**Bước 5:** Tạo biểu đồ cột để đại diện cho độ quan trọng của tính năng, được xác định bởi bộ phân loại Random Forest cho 15 tính năng hàng đầu. Điểm số độ quan trọng của một tính năng càng cao thì tính năng đó càng ảnh hưởng đến việc dự đoán với mô hình Random Forest.

A screen shot of a graph

Description automatically generated

**Bước 6**: Vẽ cây quyết định Trong đó, ta sử dụng thuộc tính estimators\_ của mô hình Random Forest để lấy ra cây quyết định đầu tiên. Sau đó, ta sử dụng hàm plot\_tree để vẽ cây quyết định với các tham số tương tự như khi sử dụng với Decision Tree. Cuối cùng, ta sử dụng hàm show() của matplotlib để hiển thị cây quyết định.

A diagram of a diagram

Description automatically generated with medium confidence

## Naïve Bayes



### Giới thiệu thuật toán Navie Bayes

Thuật toán Naïve Bayes là một kỹ thuật phân loại dựa trên việc áp dụng định lý Bayes với một giả định mạnh mẽ rằng tất cả các yếu tố dự đoán là độc lập với nhau. Nói một cách đơn giản, giả định là sự hiện diện của một đặc trưng trong một lớp là độc lập với sự hiện diện của bất kỳ đặc trưng nào khác trong cùng một lớp.

Ví dụ, một chiếc điện thoại có thể được coi là thông minh nếu nó có màn hình cảm ứng, kết nối internet, máy ảnh tốt, v.v. Mặc dù tất cả các đặc trưng này phụ thuộc lẫn nhau, chúng đóng góp một cách độc lập vào xác suất rằng điện thoại là điện thoại thông minh.

Trong phân loại Bayesian, mối quan tâm chính là tìm xác suất hậu nghiệm, tức là xác suất của một nhãn cho trước một số đặc trưng quan sát được, 𝑃(𝐿 | 𝑓𝑒𝑎𝑡𝑢𝑟𝑒𝑠). Với sự trợ giúp của định lý Bayes, chúng ta có thể biểu diễn điều này dưới dạng định lượng như sau:

* 𝑃(𝐿 | 𝑓𝑒𝑎𝑡𝑢𝑟𝑒𝑠) là xác suất của nhãn L cho trước các đặc trưng quan sát được (posterior probability).
* 𝑃(𝐿) là xác suất tiên nghiệm của nhãn L (prior probability).
* 𝑃(𝑓𝑒𝑎𝑡𝑢𝑟𝑒𝑠 | 𝐿) là xác suất của các đặc trưng cho trước nhãn L (likelihood).
* 𝑃(𝑓𝑒𝑎𝑡𝑢𝑟𝑒𝑠) là xác suất của các đặc trưng (evidence hoặc marginal likelihood).

Định lý Bayes cung cấp cách để cập nhật xác suất tiên nghiệm thành xác suất hậu nghiệm, dựa trên các thông tin mới được cung cấp bởi các đặc trưng quan sát được. Trong bối cảnh phân loại Naïve Bayes, chúng ta quan tâm đến việc tính toán P(L∣features) cho mỗi nhãn L có thể có và chọn nhãn có xác suất cao nhất làm nhãn dự đoán.

### Lý do chọn thuật toán

* Vì đây là một thuật toán phân loại dữ liệu, nên nó phù hợp với vấn đề đã đặt ra và với các trường mô tả văn bản.
* Thuật toán cho độ chính xác cao và thời gian chạy nhanh.

### Xây dựng và dự đoán với mô hình

**Bước 1:** Sử dụng hàm GaussianNB để huấn luyện và dự đoán Mức độ nghiêm trọng.

A screenshot of a computer program

Description automatically generated

* Điểm số độ chính xác trên dữ liệu huấn luyện là 0.538, cho thấy mô hình đã dự đoán chính xác nhãn lớp cho 53.8% các ví dụ trong tập huấn luyện.
* Điểm số độ chính xác trên dữ liệu xác thực là 0.537, cho thấy mô hình đã dự đoán chính xác nhãn lớp cho 53.7% các ví dụ trong tập xác thực.

**Bước 2:**

* Sử dụng mô hình Naïve Bayes để dự đoán nhãn của tập dữ liệu xác thực (X\_validate) và tính toán độ chính xác (accuracy) và điểm F1 của các dự đoán.
* Điểm độ chính xác và điểm F1 sau đó được lưu trữ trong hai từ điển (accuracy và f1) để sử dụng sau này.
* Hàm classification\_report sau đó được sử dụng để in ra độ chính xác (precision), độ thu hồi (recall), và điểm F1 cho mỗi lớp trong tập dữ liệu huấn luyện và xác thực.

A screenshot of a computer

Description automatically generated

**Bước 3:** Tính toán ma trận nhầm lẫn và vẽ nó trên biểu đồ nhiệt (heatmap).

A screenshot of a computer

Description automatically generated

**Bước 4:** Vẽ đường cong Precision-Recall (PR) cho mô hình Logistic Regression trên tập dữ liệu xác thực, đây là một cách để đánh giá hiệu suất của mô hình phân loại.

A graph with blue lines

Description automatically generated

**Bước 5**: Tương tự GaussianNB ta làm với MultinomialNB và BernoulliNB để kiểm tra

* MultinomialNB:

A chart with blue squares and white text

Description automatically generated

* BernoulliNB:

A chart with numbers and a number in a row

Description automatically generated with medium confidence

* **Như chúng ta thấy hàm gaussiannb cho kết quả nhất.**

# PHƯƠNG PHÁP ĐÁNH GIÁ VÀ DỰ ĐOÁN

## Đánh giá các thuật toán bằng Accuracy

### Giới thiệu phương pháp đánh giá

* Accuracy là một trong những phương pháp phổ biến nhất để đánh giá hiệu suất của các mô hình máy học. Accuracy có thể được tính bằng tỷ lệ giữa số dự đoán đúng trên tổng số dự đoán:

Ảnh có chứa văn bản, Phông chữ, ảnh chụp màn hình, màu trắng

Mô tả được tạo tự động

Hoặc cũng có thể bằng công thức bên dưới trong ma trận nhầm lẫn:

Ảnh có chứa văn bản, Phông chữ, màu trắng, ảnh chụp màn hình

Mô tả được tạo tự động

* Accuracy là một phương pháp đơn giản và dễ hiểu để đánh giá mức độ chính xác của mô hình máy học. Tuy nhiên, nó cũng có một số một số hạn chế, đặc biệt trong tập dữ liệu mất cân bằng.

### Lý do lựa chọn phương pháp

* Đơn giản và dễ hiều: Accuracy là một phương pháp dễ tính toán và dễ hiểu, phù hợp cho việc so sánh nhanh nhiều các thuật toán
* Phổ biến: là một phương pháp được sử dụng nhiều trong nghiên cứu và ứng dụng thực tế.
* Khả năng tổng quả lớn: accuracy sẽ đánh giá hiệu suất tổng quát của mô hình.

### Tính toán

* Decision tree

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ, hàng

Mô tả được tạo tự động

* Gaussian Naive Bayes

Ảnh có chứa văn bản, Phông chữ, ảnh chụp màn hình, hàng

Mô tả được tạo tự động

* Multinomial Naive Bayes

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ, hàng

Mô tả được tạo tự động

* Bernoulli Naive Bayes

Ảnh có chứa văn bản, Phông chữ, ảnh chụp màn hình, hàng

Mô tả được tạo tự động

* Random forest

Ảnh có chứa văn bản, Phông chữ, ảnh chụp màn hình, hàng

Mô tả được tạo tự động

* KNN

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ, hàng

Mô tả được tạo tự động

Nhận xét:

* Random forest và decision tree là 2 thuật toán có hiệu suất tổng quát tốt nhất trên tập dữ liệu
* KNN cũng cho kết quả khả quan với accuracy = 0.8439
* Các thuật toán Naïve Bayes đều cho độ chính xác khá thấp

## Phương pháp đánh giá F1 – score

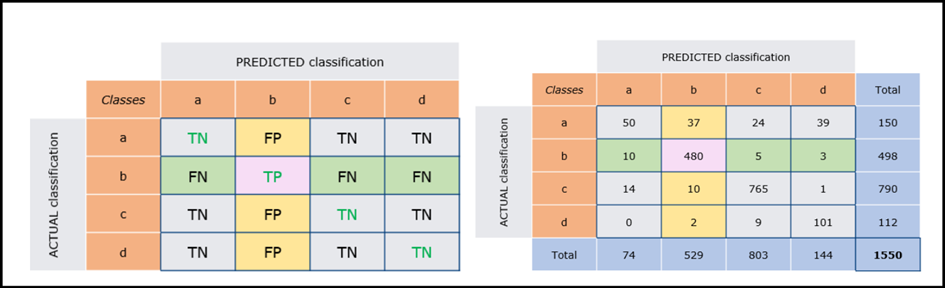
### Giới thiệu phương pháp đánh giá F1 – score

F1-Score đánh giá hiệu suất của mô hình phân loại dựa trên ma trận nhầm lẫn.

**A screenshot of a computer

Description automatically generated**

F1-Score là trung bình điều hòa của độ chính xác (precision) và độ nhạy (recall), được tính cho một lớp duy nhất trong một vấn đề phân loại nhị phân hoặc đa lớp. Trong trường hợp phân loại đa lớp, F1-Score nên xem xét tất cả các lớp. Để đạt được điều này, chúng ta cần sử dụng các thước đo đa lớp của độ chính xác và độ nhạy, sau đó kết hợp chúng bằng trung bình điều hòa. Có hai đặc tả khác nhau của các thước đo này, dẫn đến hai chỉ số đánh giá khác biệt: Micro F1-Score và Macro F1-Score. Ở đây, chúng ta sử dụng Macro F1-Score để đánh giá hiệu suất của mô hình phân loại. Cách tiếp cận Macro xem xét tất cả các lớp như là các yếu tố cơ bản trong việc tính toán: mỗi lớp được coi trọng như nhau trong trung bình cộng, bất kể kích thước của nó, do đó tránh sự phân biệt giữa các lớp có kích thước lớn và nhỏ. Để đạt được Macro F1-Score, cần có hai bước trung gian: Macro-Precision và Macro-Recall. Macro-Precision được tính bằng cách trung bình cộng các giá trị độ chính xác cho mỗi lớp được dự đoán, trong khi Macro-Recall được tính bằng cách trung bình cộng các giá trị độ nhạy cho mỗi lớp thực tế.

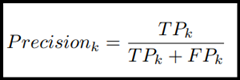


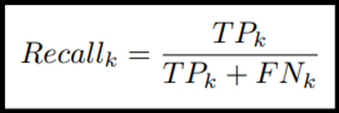
Hình ảnh trên tập trung vào lớp b làm lớp tham chiếu và minh họa cách tính Precision và Recall của nó. Để thực hiện các phép tính này, chúng ta sử dụng Ma trận nhầm lẫn (Confusion Matrix), trong đó mỗi ô được dán nhãn theo lớp tương ứng của nó. Cụ thể, chúng ta chỉ xem xét:

* True Positive (TP) là số lượng các đơn vị thuộc lớp b được mô hình phân loại chính xác là lớp b.
* False Positive (FP) là số lượng các đơn vị không thuộc lớp b nhưng bị mô hình phân loại nhầm là lớp b.
* False Negative (FN) là Số lượng các đơn vị thuộc lớp b nhưng bị mô hình phân loại nhầm thành các lớp khác.
* True Negative (TN) là Số lượng các đơn vị không thuộc lớp b và cũng không bị mô hình phân loại nhầm là lớp b.

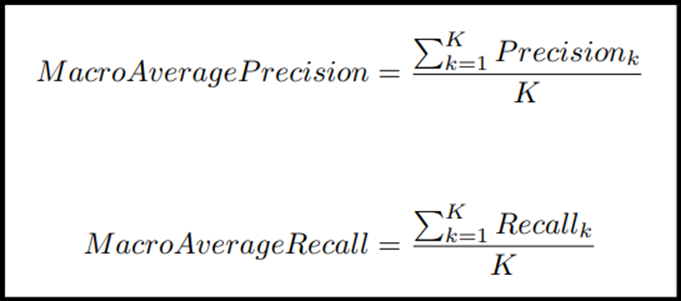
Khi chúng ta chuyển từ lớp này sang lớp khác, chúng ta lặp lại các phép tính cho Precision và Recall bằng cách sử dụng lớp tham chiếu mới và cập nhật nhãn cho các ô trong ma trận nhầm lẫn tương ứng.

Giá trị Precision và Recall cho mỗi lớp được tính bằng cách sử dụng các phương trình tương tự như trong thiết lập phân loại nhị phân, như đã mô tả trước đó.

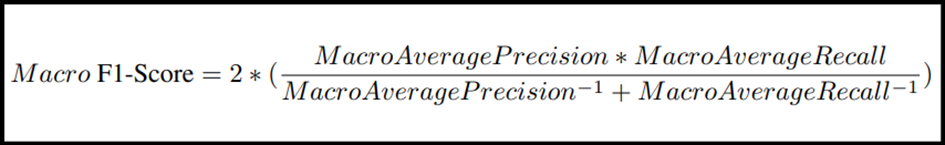




Giá trị Trung bình Macro của Precision và Recall được tính bằng cách lấy trung bình đơn giản của các giá trị Precision và Recall cho từng lớp riêng lẻ.



Cuối cùng, Macro F1-Score được tính bằng cách lấy trung bình điều hòa của các giá trị Macro-Precision và Macro-Recall:



Đồ thị PR (Precision-Recall) là một đồ thị với các giá trị Precision trên trục y và các giá trị Recall trên trục x. Nói cách khác, đồ thị PR chứa TP/(TP+FP) trên trục y và TP/(TP+FN) trên trục x.

### Lí do chọn phương pháp

Macro F1-Score được sử dụng phổ biến trong các bài toán phân loại đa lớp, đặc biệt khi dữ liệu bị mất cân bằng. Dưới đây là một số lý do cụ thể cho việc chọn phương pháp Macro F1-Score:

1. Cân bằng giữa các lớp:

* Trọng số bằng nhau: Mỗi lớp được coi trọng như nhau khi tính toán Macro F1-Score, bất kể kích thước của lớp đó. Điều này giúp đảm bảo rằng các lớp nhỏ không bị bỏ qua và các lớp lớn không chi phối kết quả đánh giá.
* Không thiên vị: Phương pháp này giúp tránh việc mô hình bị thiên vị bởi các lớp có số lượng mẫu lớn hơn, vì nó xem xét từng lớp một cách độc lập và đồng đều.

2. Đánh giá toàn diện:

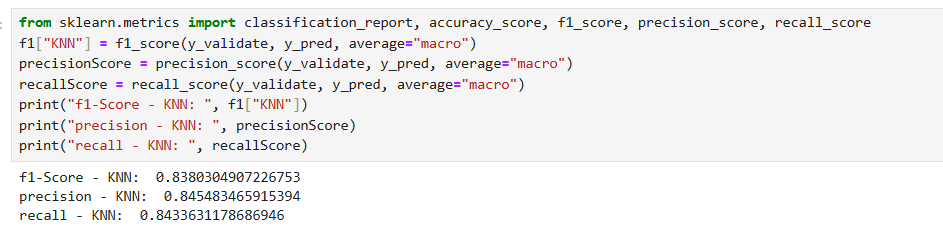
* Hiệu suất toàn bộ các lớp: Macro F1-Score cung cấp một cái nhìn tổng quan về hiệu suất của mô hình trên tất cả các lớp. Một giá trị Macro F1-Score cao cho thấy mô hình hoạt động tốt trên toàn bộ các lớp, trong khi giá trị thấp cho thấy có một hoặc nhiều lớp mà mô hình dự đoán kém.
* Phù hợp với dữ liệu mất cân bằng: Khi dữ liệu mất cân bằng, một số lớp có thể có rất ít mẫu so với các lớp khác. Macro F1-Score giúp đảm bảo rằng hiệu suất trên các lớp ít mẫu cũng được đánh giá và cải thiện.

3.Kết hợp độ chính xác và độ nhạy:

* Trung bình điều hòa: Bằng cách sử dụng trung bình điều hòa của độ chính xác (precision) và độ nhạy (recall), Macro F1-Score giúp cân bằng giữa hai yếu tố này. Điều này rất quan trọng khi một mô hình cần phải không chỉ chính xác trong các dự đoán của mình mà còn phải đảm bảo rằng hầu hết các mẫu dương tính đều được nhận diện.
* Giảm thiểu ảnh hưởng của giá trị cực đoan: Trung bình điều hòa giảm thiểu ảnh hưởng của các giá trị cực đoan (cực cao hoặc cực thấp) trong các thước đo độ chính xác và độ nhạy, giúp đánh giá chính xác hơn về hiệu suất mô hình.

### Tính toán Độ chính xác Macro, Recall và F1 – score

* KNN:



* Decision Tree:



* Gaussian Naïve Bayes

A screenshot of a computer code

Description automatically generated

* Multinomial Naïve Bayes

A screenshot of a computer code

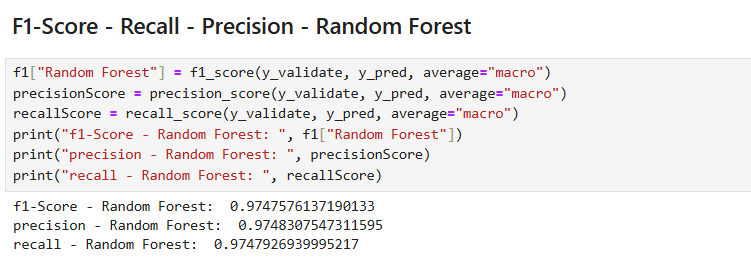
Description automatically generated

* Bernoulli Naïve Bayes

A screenshot of a computer code

Description automatically generated

* Random Forest



## Phương pháp ROC

### Giới thiệu

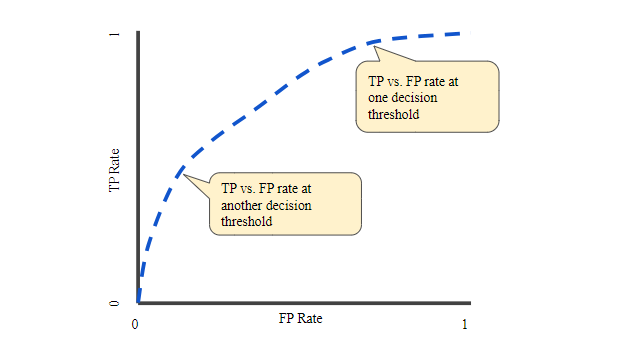
Một đồ thị ROC (Receiver Operating Characteristic Curve) là là một biểu đồ cho thấy hiệu suất của mô hình phân loại ở tất cả các ngưỡng phân loại. Đường cong này hiển thị hai tham số:

* True Positive Rate (TPR) hay Sensitivity/Recall: TPR = TP / (TP + FN), đại diện cho tỷ lệ các mẫu dương tính thực sự được dự đoán đúng.
* False Positive Rate (FPR): FPR = FP / (FP + TN), đại diện cho tỷ lệ các mẫu âm tính thực sự bị dự đoán sai.

Trong đó:

* TP: Dương tính thật (dự đoán dương tính trong các trường hợp thực sự dương tính)
* FN: Âm tính giả (dự đoán âm tính trong các trường hợp thực sự dương tính)
* FP: Dương tính giả (dự đoán dương tính trong các trường hợp thực sự âm tính)
* TN: Âm tính thật (dự đoán âm tính trong các trường hợp thực sự âm tính)

Đường cong ROC vẽ đồ thị TPR so với FPR ở các ngưỡng phân loại khác nhau. Việc giảm ngưỡng phân loại sẽ phân loại nhiều mục hơn là dương, do đó tăng cả số dương tính giả và dương tính thực. Hình bên dưới cho thấy đường cong ROC điển hình.



Các đồ thị ROC thường hiển thị tỷ lệ dương tính thật (True Positive Rate) trên trục Y và tỷ lệ dương tính giả (False Positive Rate) trên trục X. Điều này có nghĩa là góc trên bên trái của biểu đồ là điểm "lý tưởng" - với tỷ lệ dương tính giả bằng không và tỷ lệ dương tính thật bằng một.

A white rectangle with blue border

Description automatically generated with low confidence

Điều này không thực tế lắm, nhưng nó có nghĩa là diện tích dưới đường cong (AUC) lớn hơn thường tốt hơn.

### Lý do chọn phương pháp

* Đánh giá toàn diện ở các ngưỡng khác nhau:
  + ROC Curve cho phép đánh giá hiệu suất của mô hình ở tất cả các ngưỡng phân loại khác nhau, không chỉ ở một ngưỡng cố định. Điều này giúp có cái nhìn tổng quan về khả năng phân loại của mô hình.
* So sánh các mô hình:
  + Diện tích dưới đường cong ROC (AUC) là một chỉ số tóm tắt cho phép so sánh trực quan và dễ dàng giữa các mô hình khác nhau. Mô hình nào có AUC lớn hơn thì mô hình đó có hiệu suất tốt hơn.
* Không phụ thuộc vào tỷ lệ dữ liệu không cân bằng:
  + ROC Curve không bị ảnh hưởng bởi tỷ lệ lớp không cân bằng trong dữ liệu, vì nó dựa trên tỷ lệ chứ không phải là số lượng tuyệt đối. Điều này rất hữu ích khi làm việc với các tập dữ liệu mà một lớp có tỷ lệ nhỏ hơn rất nhiều so với các lớp khác.
* Cân nhắc giữa True Positive Rate và False Positive Rate:
  + ROC Curve giúp hiểu rõ hơn về trade-off giữa Tỷ lệ dương tính thật (True Positive Rate) và Tỷ lệ dương tính giả (False Positive Rate). Điều này quan trọng trong các ứng dụng mà cả hai chỉ số này đều cần được tối ưu hóa.
* Lựa chọn ngưỡng tối ưu:
  + Bằng cách quan sát ROC Curve, bạn có thể lựa chọn ngưỡng phân loại tối ưu cho mô hình, dựa trên các yêu cầu cụ thể của bài toán (ví dụ: giảm thiểu False Positives hoặc tối đa hóa True Positives).
* Ứng dụng đa dạng:
  + ROC Curve không chỉ áp dụng cho các bài toán nhị phân mà còn có thể mở rộng cho các bài toán phân loại đa nhãn thông qua các phương pháp như nhị phân hóa đầu ra hoặc micro-averaging.
* Dễ hiểu và trực quan:
  + ROC Curve là một công cụ trực quan, dễ hiểu và dễ sử dụng, ngay cả đối với những người không chuyên về phân tích dữ liệu hoặc học máy.

Với những lý do trên, ROC Curve là một công cụ mạnh mẽ và linh hoạt để đánh giá và so sánh hiệu suất của các mô hình dự đoán, đặc biệt trong các bài toán phân loại.

### Tính toán

* KNN:

A person holding a microphone

Description automatically generated with medium confidence A graph of a person

Description automatically generated with medium confidence

* Decision tree:



A graph with a line

Description automatically generated with medium confidence

* Random forest

 A graph with a blue line

Description automatically generated

* Navie Bayes

A screen shot of a computer

Description automatically generated

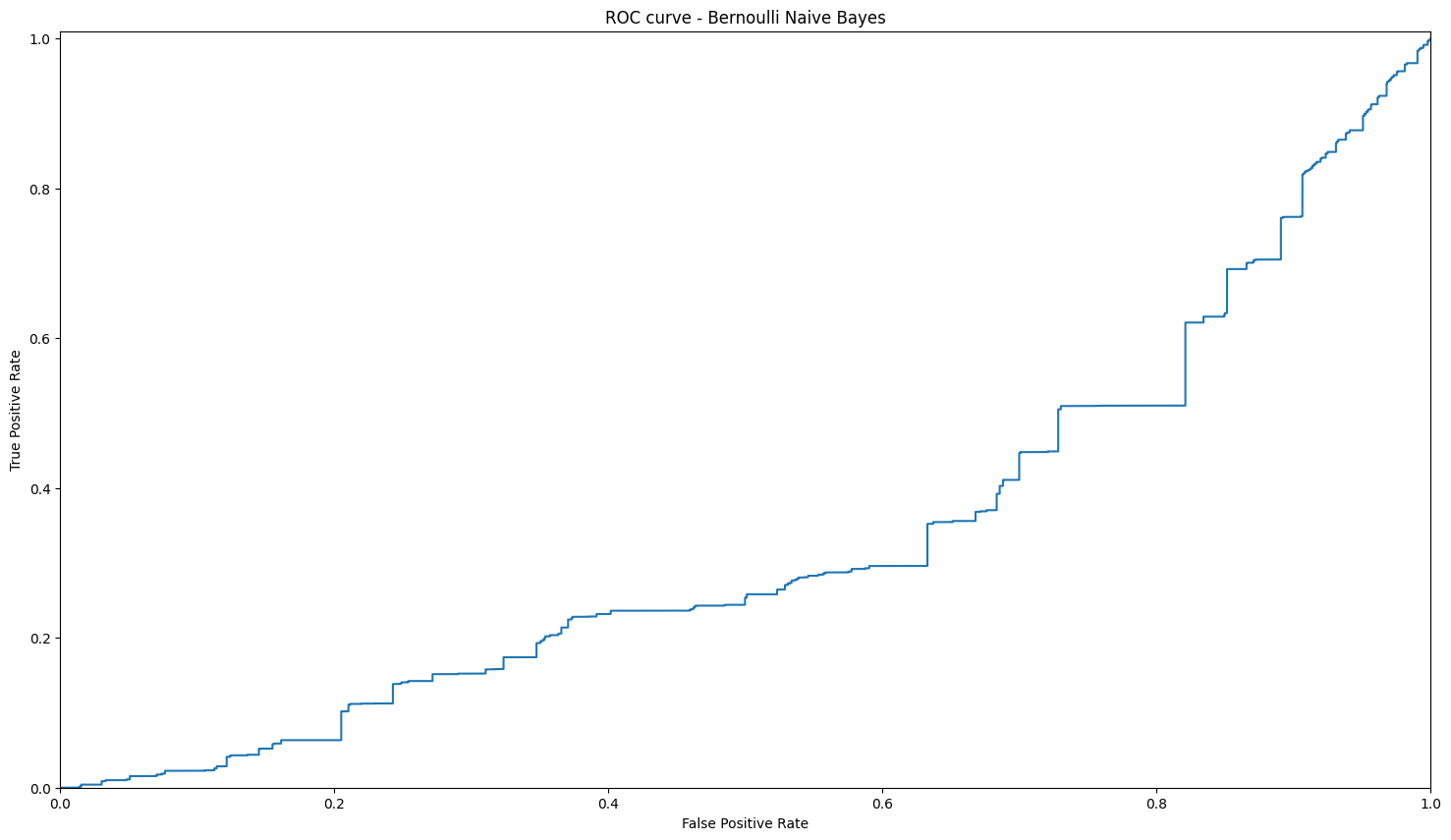
A graph with a line

Description automatically generated

A screenshot of a computer

Description automatically generated A graph of a growing graph

Description automatically generated with medium confidence A screen shot of a computer

Description automatically generated 

## Dự đoán

* So sánh giá trị Accuracy, F1-Score, biểu đồ PR curve, ROC curve giữa các mô hình thuật toán:

**A screenshot of a graph

Description automatically generated**

**A screenshot of a graph

Description automatically generated**

**A graph of a graph

Description automatically generated with medium confidence**

**A graph of a graph

Description automatically generated with medium confidence**

Dựa vào các kết quả đánh giá thuật toán trên, ta thấy thuật toán tối ưu. có độ ổn định và phù hợp với bộ dữ liệu trên nhất chính là thuật toán Random Forest.

Nên tiến hành dự đoán bằng RF:

**A screenshot of a computer screen

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

1. **TỔNG KẾT**

## Ưu điểm

- Việc áp dụng nhiều thuật toán phân loại giúp nhóm có cái nhìn đa dạng hơn về dữ liệu và giúp phân loại tốt hơn, chính xác hơn.

- Các thuật toán phân loại dễ dàng cho phép đánh giá hiệu quả hoạt động của mô hình thông qua các thước đo như accuracy, precision, recall, F1-score, ROC curve, PR curve, confusion matrix.

## Hạn chế

* Các thuật toán khó, mất nhiều thời gian để tìm hiểu.
* Chưa khai thác tối đa tiềm năng của bộ dữ liệu.

## Kết quả đạt được

* Trong quá trình triển khai dự án, nhóm đã tích lũy được rất nhiều kiến thức liên quan đến khai thác dữ liệu trên một bộ dữ liệu cụ thể: tiền xử lý, phân lớp, dự đoán bằng các thuật toán đã học. Sau khi thảo luận, lập kế hoạch và phân chia công việc thực hiện thành một dự án hoàn chỉnh với từng mô hình.
* Việc này giúp nhóm có được hiểu biết sâu hơn về cách xử lý các bộ dữ liệu hiện có và cách hoạt động của các thuật toán đã sử dụng.

## Định hướng phát triển

Trong tương lai, nhóm có kế hoạch thực hiện những cải tiến sau đây:

* Nghiên cứu thêm các thuật toán khác để phù hợp hơn với bộ dữ liệu, đưa ra các dự đoán, phân lớp chính xác và đạt hiệu quả tối ưu.
* Cải thiện tốc độ truy xuất kết quả dự báo.
* Nghiên cứu và tiền xử lý dữ liệu tốt hơn để giảm tài nguyên tính toán và thời gian huấn luyện, dự đoán.

# DANH MỤC THAM KHẢO

|  |  |
| --- | --- |
| [1] | Developers.google.com, "Machine Learning," 2022-09-26 UTC.. [Online]. Available: https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc?hl=vi. |