



TRƯỜNG ĐẠI HỌC  
**SƯ PHẠM KỸ THUẬT TP. HỒ CHÍ MINH.**  
HCMC University of Technology and Education



## **BÁO CÁO HỌC PHẦN**

**TÊN HỌC PHẦN: ĐIỆN TOÁN ĐÁM MÂY**

**MÃ SỐ LỚP HP: 211CLCO332779 – (Tối thứ 3)**

**TÊN ĐỀ TÀI: TÌM HIỂU AWS SAGEMAKER  
VÀ VIẾT ỨNG DỤNG MINH HỌA**

**GVHD: TS. Huỳnh Xuân Phụng**

Họ tên sinh viên [1]: **DIỆP THÁI DƯƠNG 19110009**

Họ tên sinh viên [2]: **NGUYỄN KỲ HẢI 19110197**

Họ tên sinh viên [3]: **TRẦN NGUYỄN QUỐC KHÁNH 19110227**

**Tp.Hồ Chí Minh, ngày 30 tháng 11 năm 2021**



## BÁO CÁO HỌC PHẦN

**Giảng viên hướng dẫn: TS. Huỳnh Xuân Phụng**

**TÊN HỌC PHẦN: ĐIỆN TOÁN ĐÁM MÂY**

**MÃ SỐ LỚP HP: 211CLCO332779 – (Tối thứ 3)**

**TÊN ĐỀ TÀI: TÌM HIỂU AWS SAGEMAKER VÀ VIẾT ỨNG DỤNG MINH HỌA**

Họ tên sinh viên [1]: **DIỆP THÁI DƯƠNG 19110009**

Họ tên sinh viên [2]: **NGUYỄN KỲ HẢI 19110197**

Họ tên sinh viên [3]: **TRẦN NGUYỄN QUỐC KHÁNH 19110227**

*Nhận xét của giảng viên*

.....  
.....  
.....

**Điểm đánh giá: .....(.....)**

*TP.HCM, ngày....tháng....năm 2021*

Giảng viên chấm điểm

*(Ký tên)*

## LỜI CẢM ƠN

Để hoàn thành tốt đề tài và bài báo cáo này, chúng em xin gửi lời cảm ơn chân thành đến giảng viên, **TS Huỳnh Xuân Phụng**, người đã trực tiếp hỗ trợ chúng em trong suốt quá trình làm đề tài. Chúng em cảm ơn thầy đã đưa ra những lời khuyên từ kinh nghiệm thực tiễn của mình để định hướng cho chúng em đi đúng với yêu cầu của đề tài đã chọn, luôn giải đáp thắc mắc và đưa ra những góp ý, chỉnh sửa kịp thời giúp chúng em khắc phục nhược điểm và hoàn thành tốt cũng như đúng thời hạn đã đề ra.

Chúng em cũng xin gửi lời cảm ơn chân thành các quý thầy cô trong khoa Đào tạo Chất Lượng Cao nói chung và ngành Công Nghệ Thông Tin nói riêng đã tận tình truyền đạt những kiến thức cần thiết giúp chúng em có nền tảng để làm nên đề tài này, đã tạo điều kiện để chúng em có thể tìm hiểu và thực hiện tốt đề tài. Cùng với đó, chúng em xin được gửi cảm ơn đến các bạn cùng khóa đã cung cấp nhiều thông tin và kiến thức hữu ích giúp chúng em có thể hoàn thiện hơn đề tài của mình.

Đề tài và bài báo cáo được chúng em thực hiện trong khoảng thời gian ngắn, với những kiến thức còn hạn chế cùng nhiều hạn chế khác về mặt kĩ thuật và kinh nghiệm trong việc thực hiện một dự án phần mềm. Do đó, trong quá trình làm nên đề tài có những thiếu sót là điều không thể tránh khỏi nên chúng em rất mong nhận được những ý kiến đóng góp quý báu của các quý thầy cô để kiến thức của chúng em được hoàn thiện hơn và chúng em có thể làm tốt hơn nữa trong những lần sau.

Chúng em xin chân thành cảm ơn. Cuối lời, chúng em kính chúc quý thầy, quý cô luôn dồi dào sức khỏe và thành công hơn nữa trong sự nghiệp trồng người. Một lần nữa chúng em xin chân thành cảm ơn.

Nhóm sinh viên thực hiện.

# MỤC LỤC

CHƯƠNG 1 GIỚI THIỆU CHUNG VỀ ĐỀ TÀI .....	1
1.1 Lý do chọn đề tài.....	1
1.2 Tổng quan về đề tài .....	1
1.3 Nội dung chuyên môn chính của đề tài.....	2
1.4 Mô tả ứng dụng .....	2
CHƯƠNG 2 CƠ SỞ LÝ THUYẾT CỦA ĐỀ TÀI.....	4
2.1 Tổng quan về Điện toán đám mây (Cloud Computing): .....	4
2.2 Tổng quan về AWS.....	4
2.3 Tổng quan về S3 .....	5
2.4 Tổng quan về SageMaker .....	6
CHƯƠNG 3 PHÂN TÍCH VÀ THIẾT KẾ ĐỀ TÀI .....	7
3.1 Hướng dẫn sử dụng Amazon SageMaker Studio .....	7
3.2 Hướng dẫn các bước thực hiện dự án mô hình học máy .....	8
3.2.1 Chuẩn bị dữ liệu và tạo Notebook .....	8
3.2.2 Xử lý và làm sạch dữ liệu .....	10
3.2.3 Phân tách tập dữ liệu.....	15
3.2.4 Đào tạo mô hình dùng SageMaker .....	16
3.2.5 Triển khai mô hình.....	20
3.2.6 Kiểm tra.....	21
CHƯƠNG 4 TỔNG KẾT .....	25
4.1 Kết quả đạt được: .....	25
4.2 Những hạn chế: .....	25
4.3 Hướng phát triển: .....	25
TÀI LIỆU THAM KHẢO .....	26

## DANH SÁCH HÌNH ẢNH

Hình 3: Quy trình xây dựng hệ thống máy học dùng AWS SageMaker .....	7
Hình 3.1 a: Giao diện Amazon SageMaker .....	7
Hình 3.1 b: Chọn IDE và chuyển hướng đến giao diện phát triển mô hình ML ..	8
Hình 3.1 c: Giao diện môi trường JupyterLab .....	8
Hình 3.2.1 a: Tập dữ liệu insurance.csv.....	9
Hình 3.2.1 b: Tạo file notebook .....	9
Hình 3.2.1 c: Thiết lập môi trường notebook .....	10
Hình 3.2.2 a: Đọc tập tin insurance.csv .....	11
Hình 3.2.2b: Một số thông tin về dữ liệu.....	12
Hình 3.2.2 c: Kết quả kiểm tra giá trị null của tập dữ liệu .....	12
Hình 3.2.2 d: Kết quả chuyển đổi dữ liệu ở cột sex của tập dữ liệu.....	12
Hình 3.2.2 e: Kết quả chuyển đổi dữ liệu ở cột smoker .....	13
Hình 3.2.2 g: Kết quả sau khi chuyển đổi cột region .....	13
Hình 3.2.2 h: Kết quả sau khi xử lý tập dữ liệu .....	14
Hình i: Mối quan hệ giữa age và charge .....	14
Hình j: Mối quan hệ giữa bmi và charge .....	15
Hình 3.2.3 a: Loại bỏ cột charge đầu vào và thêm vào cột charge đầu ra.....	15
Hình 3.2.3 b : Chuyển đổi dữ liệu X, y.....	16
Hình 3.2.3 c: Kết quả phân tách tập dữ liệu .....	16
Hình 3.2.3 d: Kết quả phân trăm độ chính xác của mô hình .....	16
Hình 3.2.4 a: Kết quả vai trò IAM của SageMaker .....	17
Hình 3.2.4 b: Chuyển đổi sang dạng vector.....	17
Hình 3.2.4 c: Chuyển đổi dữ liệu đào tạo .....	18
Hình 3.2.4 e: Kết quả vị trí nơi lưu trữ dữ liệu đào tạo .....	18
Hình 3.2.4. g: Kết quả vị trí lưu trữ dữ liệu đầu ra .....	18
Hình 3.2.4 h: Kết quả được tạo ra ở S3 .....	19
Hình 3.2.4 i: Chỉ định tên thuật toán để đào tạo .....	19
Hình 3.2.4 j: Chuẩn bị trước khi đào tạo mô hình .....	19
Hình 3.2.4 k: Các tham số đào tạo mô hình.....	20
Hình 3.2.4 m: Kết quả sau khi đào tạo.....	20

Hình 3.2.5 a: Kết quả khi triển khai mô hình .....	20
Hình 3.2.5 b: Điểm truy cập end point .....	21
Hình 3.2.6 a : Kết quả dự đoán .....	22
Hình 3.2.6 b: Kết quả dưới dạng array .....	23
Hình 3.2.6 c: Kết quả thực tế và kết quả dự đoán.....	24
Hình 3.2.6 d: Kết quả dưới dạng biểu đồ.....	24
Hình 3.2.6 e: Xóa điểm cuối end point .....	24

# CHƯƠNG 1 GIỚI THIỆU CHUNG VỀ ĐỀ TÀI

## 1.1 Lý do chọn đề tài

Ngày nay, học máy (Machine Learning) là một lĩnh vực của trí tuệ nhân tạo liên quan đến việc nghiên cứu và xây dựng các kỹ thuật cung cấp cho các hệ thống khả năng tự học và cải thiện kinh nghiệm từ dữ liệu đầu vào để giải quyết những vấn đề cụ thể.

AWS là nền tảng đám mây toàn diện và được sử dụng rộng rãi nhất, cung cấp trên 200 dịch vụ đầy đủ tính năng từ các trung tâm dữ liệu trên toàn thế giới. Một trong những dịch vụ phát triển nhanh nhất trong lịch sử AWS là Amazon SageMaker. Amazon SageMaker được xây dựng dựa trên hai thập kỷ kinh nghiệm của Amazon trong việc phát triển các ứng dụng máy học thực tế, là một giải pháp AWS được quản lý hoàn toàn, cho phép các nhà khoa học và nhà phát triển dữ liệu nhanh chóng xây dựng, đào tạo và triển khai các mô hình học máy.

Do đó để tìm hiểu về cách thức hoạt động cũng như cách xây dựng, đào tạo và triển khai mô hình học máy sử dụng Amazon SageMaker, nhóm em đã chọn đề tài Tìm hiểu AWS SageMaker và viết ứng dụng minh họa để làm rõ các vấn đề nêu trên.

## 1.2 Tổng quan về đề tài

Đối với những người yêu công nghệ thì họ có thể sẽ đặt ra câu hỏi: “Tại sao tôi nên sử dụng AWS SageMaker?”. Để trả lời câu hỏi trên thì chúng ta cần phải biết được những khó khăn khi triển khai một dự án học máy. Mức độ phức tạp của dự án của doanh nghiệp sẽ tăng lên khi quy mô được mở rộng. Điều này là do các dự án học máy bao gồm ba giai đoạn chính: xây dựng, đào tạo và triển khai, mỗi giai đoạn có thể liên tục lặp lại, và khi dữ liệu được xử lý tăng lên, thì độ phức tạp cũng tăng theo.

Để giúp các nhà khoa học dữ liệu và nhà phát triển có thể đưa ra các mô hình học máy khả thi trên quy mô lớn với thời gian, công sức và chi phí ít hơn thì AWS SageMaker xuất hiện. Là một nền tảng học máy được quản lý hoàn toàn, SageMaker tóm tắt các kỹ năng phần mềm, cho phép các kỹ sư dữ liệu xây dựng và đào tạo các mô hình học máy mà họ muốn với một bộ công cụ trực quan và dễ sử dụng. Amazon

SageMaker đóng gói tất cả các thành phần được sử dụng cho học máy trong một lớp vỏ duy nhất, cho phép nhà khoa học dữ liệu cung cấp các dự án ML từ đầu đến cuối, với công sức giảm thiểu và chi phí thấp hơn.

### 1.3 Nội dung chuyên môn chính của đề tài

- Tìm hiểu dịch vụ lưu trữ dữ liệu của AWS: AWS S3.
- Tìm hiểu SageMaker:
  - + Cách thiết lập môi trường phát triển tích hợp (IDE) của SageMaker.
  - + Các bước để triển khai một mô hình máy học dùng SageMaker.
  - + Kiểm tra trên mô hình đã triển khai.

### 1.4 Mô tả ứng dụng

Xây dựng chương trình dự đoán chi phí bảo hiểm cho các khách hàng khác nhau dựa trên các thông tin về độ tuổi, giới tính, chỉ số BMI, thói quen hút thuốc lá và vị trí địa lý.

Chương trình sẽ nhận vào tập dữ liệu csv, sau đó dùng SageMaker thực hiện các bước để triển khai mô hình dùng SageMaker để đưa ra dự đoán về phí bảo hiểm. Dữ liệu trước và sau khi đào tạo sẽ được lưu trữ và quản lý trên AWS S3.

Data source: <https://www.kaggle.com/mirichoi0218/insurance>

#### 1.1. Phân công công việc:

S T T	Thời gian	Công việc	Người thực hiện	Kết quả
1	5/10/2021 – 12/10/2021	Tìm hiểu về dịch vụ AWS SageMaker	Cả nhóm	Hoàn thành
2	12/10/2021 – 19/10/2021	Lựa chọn, thống nhất và mô tả sơ bộ về đề tài	Cả nhóm	Hoàn thành
3	19/10/2021 – 26/10/2021	Tìm kiếm tập dữ liệu ( <a href="https://www.kaggle.com/">https://www.kaggle.com/</a> )	Dương	Hoàn thành



4	26/10/2021 – 2/11/2021	Tìm hiểu IDE SageMaker Studio	Hải	Hoàn thành
5	2/11/2021 – 9/11/2021	Tiến hành viết chương trình xử lý dữ liệu ban đầu	Dương	Hoàn thành
6	9/11/2021 – 16/11/2021	Trực quan hóa và phân tách tập dữ liệu	Khánh	Hoàn thành
7	16/11/2021 – 23/11/2021	Tiến hành đào tạo và triển khai mô hình dùng SageMaker	Khánh	Hoàn thành
8	23/11/2021 – 28/11/2021	Kiểm tra mô hình	Hải	Hoàn thành
9	28/11/2021 – 30/11/2021	Viết báo cáo	Cả nhóm	Hoàn thành

# CHƯƠNG 2 CƠ SỞ LÝ THUYẾT CỦA ĐỀ TÀI

## 2.1 Tổng quan về Điện toán đám mây (Cloud Computing):

Thuật ngữ “Điện toán đám mây” (Cloud Computing) để chỉ mô hình điện toán sử dụng các công nghệ máy tính và phát triển dựa vào mạng Internet.

Điện toán đám mây là một thị trường đang phát triển ngày một mạnh mẽ do chúng cung cấp công nghệ cho các công ty thuê, phục vụ qua kết nối Internet và chỉ phải trả cho những gì họ sử dụng. Điều này trái ngược với phương pháp truyền thống là mua phần cứng và phần mềm rồi tự cài đặt và bảo trì. Vì vậy số lượng các doanh nghiệp đang chuyển sang ứng dụng công nghệ đám mây ngày một tăng nhanh, điện toán đám mây đang dần trở thành giải pháp chủ đạo trong kinh doanh cũng như trong cuộc sống hàng ngày. Rất nhiều doanh nghiệp trên thế giới đã và đang áp dụng mô hình điện toán đám mây và ứng dụng những tính năng của hệ thống này trong doanh nghiệp của mình. Từ những lá thư điện tử mà chúng ta vẫn kiểm tra mỗi ngày cho đến bộ nhớ điện thoại, tất cả đều được lưu trữ và quản lý bởi điện toán đám mây.

Ứng dụng của điện toán đám mây không chỉ dừng lại ở các email cá nhân hay lưu trữ dữ liệu, mà nhờ khả năng mở rộng linh hoạt điện toán đám mây đã trở thành phương tiện được lựa chọn để phát triển, thử nghiệm và triển khai phần mềm. Các ví dụ về điện toán đám mây ngày nay có thể tìm thấy ở bất cứ đâu, từ các ứng dụng tin nhắn cho tới video streaming, chatbots ...

## 2.2 Tổng quan về AWS

AWS – Amazon Web Services từ lâu đã được coi là “Gã khổng lồ” trong việc triển khai nền tảng dịch vụ điện toán đám mây, khi mà thị phần của nó còn lớn hơn thị phần của 4 đối thủ kế tiếp là Microsoft, Google, IBM và Alibaba cộng lại.

AWS là nền tảng dịch vụ điện toán đám mây an toàn, mang đến khả năng tính toán, lưu trữ cơ sở dữ liệu, phân phối nội dung và các chức năng khác nhằm giúp các doanh nghiệp mở rộng và phát triển.

AWS là nền tảng đám mây toàn diện và được sử dụng rộng rãi nhất, cung cấp trên 165 dịch vụ đầy đủ tính năng từ các trung tâm dữ liệu trên toàn thế giới. Hàng triệu khách hàng—bao gồm các công ty khởi nghiệp tăng trưởng nhanh nhất, các tập đoàn lớn nhất cũng như các cơ quan hàng đầu của chính phủ—đều tin tưởng vào AWS để phát triển cơ sở hạ tầng, trở nên linh hoạt hơn và giảm chi phí.

Các dịch vụ chính mà AWS cung cấp: Tính toán, lưu trữ, phân phối mạng và nội dung, các công cụ quản lý, các công cụ phát triển, phân tích, học máy, công nghệ thực tế ảo ...

## 2.3 Tổng quan về S3

Amazon Simple Storage Service (Amazon S3) là một dịch vụ lưu trữ đối tượng cung cấp khả năng thay đổi theo quy mô, tính khả dụng của dữ liệu, bảo mật và hiệu năng hàng đầu trong lĩnh vực. Điều này có nghĩa là khách hàng thuộc mọi quy mô và lĩnh vực có thể sử dụng dịch vụ này để lưu trữ và bảo vệ bất kỳ lượng dữ liệu nào cho nhiều trường hợp sử dụng khác nhau, chẳng hạn như trang web, ứng dụng di động, sao lưu và khôi phục, lưu trữ, ứng dụng doanh nghiệp, thiết bị IoT và phân tích dữ liệu lớn.

Amazon S3 cung cấp các tính năng quản lý dễ sử dụng, nhờ đó, bạn có thể tổ chức dữ liệu và cấu hình các kiểm soát truy cập được tinh chỉnh để đáp ứng yêu cầu cụ thể của doanh nghiệp, tổ chức và yêu cầu về tuân thủ. **Amazon S3** được thiết kế để đảm bảo độ bền 99,999999999% (11,9's downtime) và lưu trữ dữ liệu của hàng triệu ứng dụng cho các công ty trên toàn thế giới.

Dữ liệu trên **Amazon S3** được hệ thống hoá theo khái niệm Bucket:

- Amazon S3 lưu trữ dữ liệu như một Object.
- Một **Bucket** là một đơn vị lưu trữ logical trên **S3**.
- Một Bucket sẽ chứa các object, trong đó Object sẽ chứa dữ liệu (data) và metadata miêu tả về dữ liệu đó.
- Chúng ta có thể chỉ định vị trí địa lý khu vực mà Bucket của bạn tạo ra sẽ được lưu trữ ở **Amazon S3** khu vực đó.

- Mỗi bucket sẽ có một cái tên toàn cục (global) độc nhất. Điều đó có nghĩa nếu tên bucket đã được xài thì không tài khoản AWS nào ở bất kì khu vực nào có thể đặt lại tên đó.

## 2.4 Tổng quan về SageMaker

Amazon SageMaker là một dịch vụ học máy được quản lý hoàn toàn. Với Amazon SageMaker, các nhà khoa học và nhà phát triển dữ liệu có thể nhanh chóng và dễ dàng xây dựng và đào tạo các mô hình học máy, sau đó trực tiếp triển khai chúng vào một môi trường lưu trữ sẵn sàng sản xuất. Nó cung cấp một phiên bản sổ ghi chép tác giả Jupyter tích hợp để dễ dàng truy cập vào các nguồn dữ liệu của chúng ta để khám phá và phân tích, do đó chúng ta không phải quản lý máy chủ. Nó cũng cung cấp các thuật toán học máy phổ biến được tối ưu hóa để chạy hiệu quả với dữ liệu cực lớn trong môi trường phân tán. Với hỗ trợ nguyên bản cho các thuật toán và khuôn khổ khác nhau, Amazon SageMaker cung cấp các tùy chọn đào tạo phân tán linh hoạt điều chỉnh cho phù hợp với quy trình công việc cụ thể. Triển khai mô hình vào một môi trường an toàn và có thể mở rộng bằng cách khởi chạy mô hình đó với một cú nhấp chuột từ bảng điều khiển Amazon SageMaker.

AWS SageMaker thực sự là một phần của cuộc cách mạng không máy chủ: nhà phát triển cung cấp mã chứ không quản lý máy chủ và cơ sở hạ tầng. Do đó sẽ có ít cấu hình hơn, ít chi phí phát triển ứng dụng hơn.

Quy trình làm việc của SageMaker:

- Bắt đầu phiên bản sổ ghi chép
- Tải xuống dữ liệu
- Xử lý chuẩn bị dữ liệu, làm sạch dữ liệu
- Xuất dữ liệu CSV và tải lên S3 (bộ nhớ)
- Mô hình đào tạo với dữ liệu bên trên
- Kiểm tra mô hình được đào tạo
- Triển khai mô hình
- Sử dụng mô hình đã nói thông qua điểm cuối end-point

## CHƯƠNG 3 PHÂN TÍCH VÀ THIẾT KẾ ĐỀ TÀI

Như đã phân tích bên trên chúng ta sẽ thực hiện các bước theo quy trình làm việc của SageMaker.

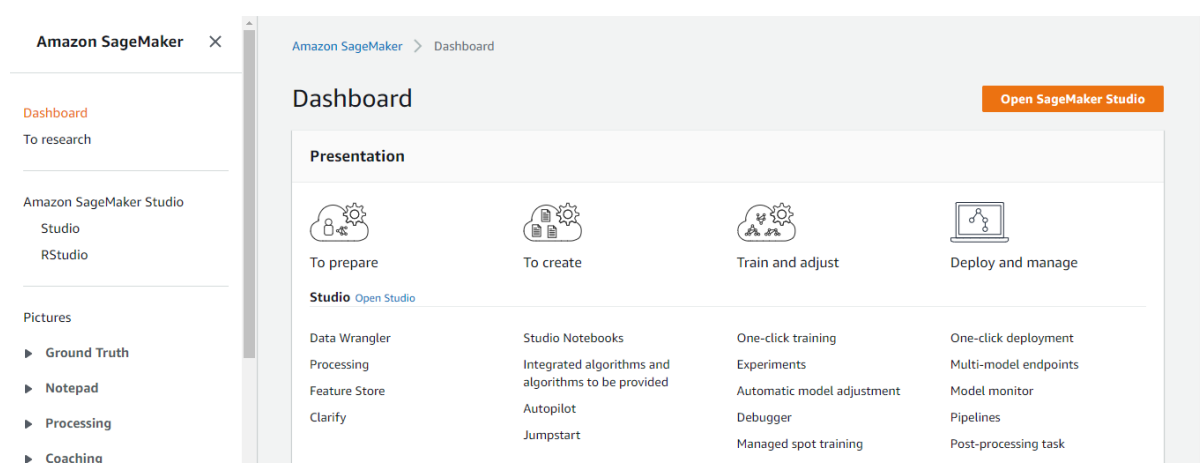


Hình 3: Quy trình xây dựng hệ thống máy học dùng AWS SageMaker

### 3.1 Hướng dẫn sử dụng Amazon SageMaker Studio

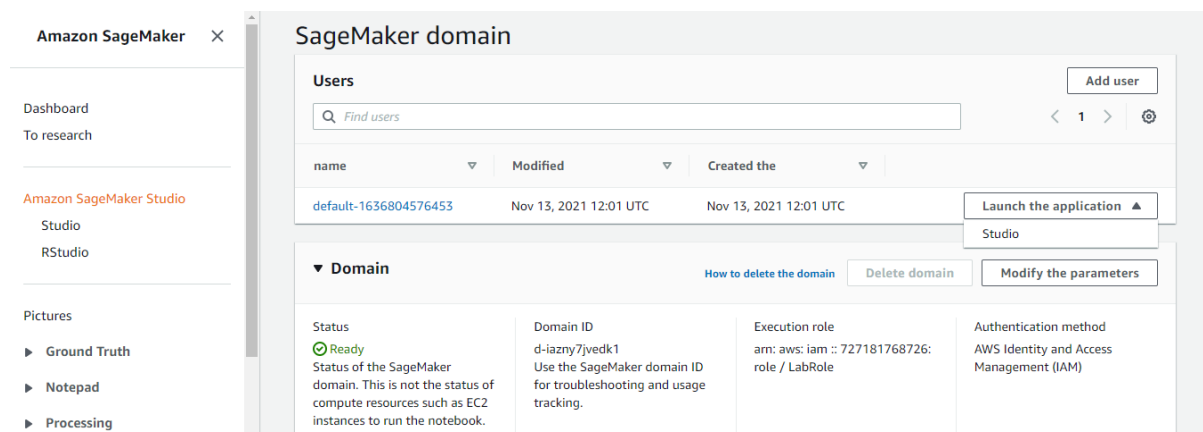
Amazon SageMaker Studio là môi trường phát triển tích hợp (IDE) đầy đủ đầu tiên được thiết kế riêng cho Machine Learning, mang lại mọi thứ chúng ta cần cho ML dưới một giao diện người dùng trực quan, thống nhất. Ngoài ra sử dụng môi trường phát triển tích hợp này còn giúp người dùng tiết kiệm được thời gian, công sức và cả chi phí.

Trên giao diện AWS Management Console, trong phần Services tìm và chọn Amazon SageMaker.

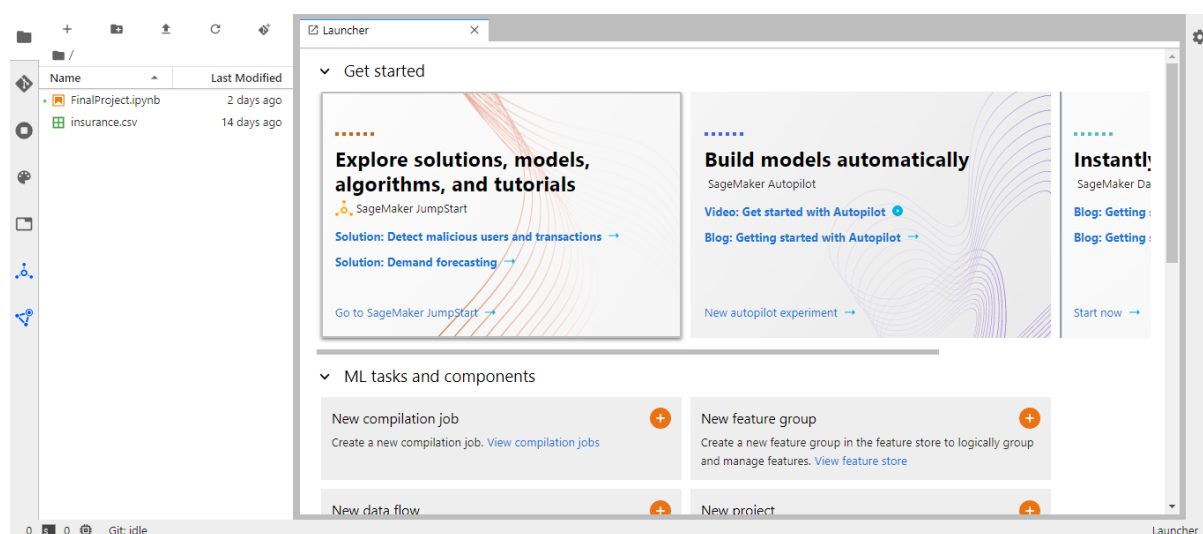


Hình 3.1 a: Giao diện Amazon SageMaker

Trên thanh điều hướng bên trái, chọn môi trường phát triển tích hợp Amazon SageMaker Studio chọn Launch the application để chuyển đến môi trường phát triển ML của SageMaker.



Hình 3.1 b: Chọn IDE và chuyển hướng đến giao diện phát triển mô hình ML



Hình 3.1 c: Giao diện môi trường JupyterLab

## 3.2 Hướng dẫn các bước thực hiện dự án mô hình học máy

### 3.2.1 Chuẩn bị dữ liệu và tạo Notebook

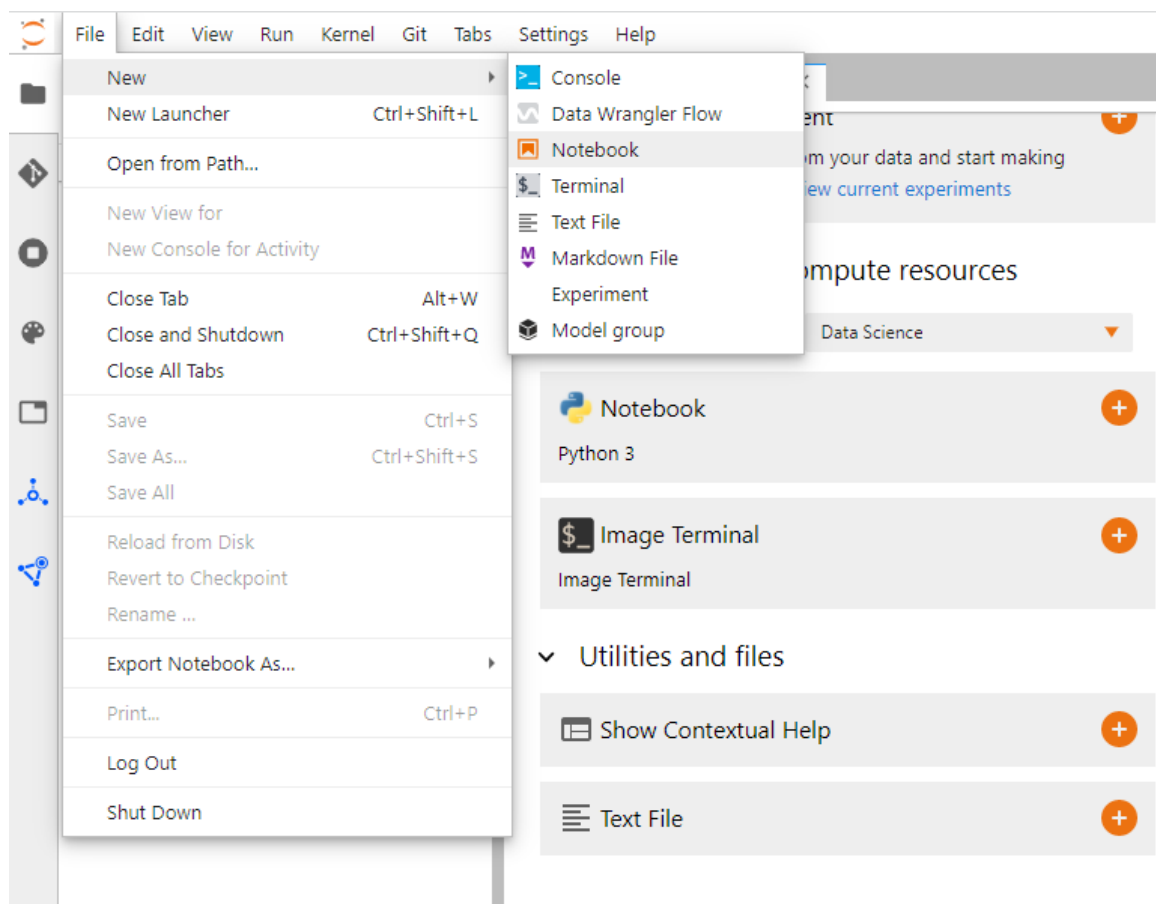
Như những mô tả bên trên thì ứng dụng này dùng để đưa ra dự đoán về chi phí bảo hiểm mà khách hàng phải trả dựa trên các thông tin về độ tuổi, giới tính, bmi, số con, thói quen hút thuốc và vị trí địa lý khác nhau.

age	sex	bmi	children	smoker	region	charges
19	female	27.9	0	yes	southwest	16884.92
18	male	33.77	1	no	southeast	1725.552
28	male	33	3	no	southeast	4449.462
33	male	22.705	0	no	northwest	21984.47
32	male	28.88	0	no	northwest	3866.855
31	female	25.74	0	no	southeast	3756.622
46	female	33.44	1	no	southeast	8240.59
37	female	27.74	3	no	northwest	7281.506
37	male	29.83	2	no	northeast	6406.411
60	female	25.84	0	no	northwest	28923.14

*Hình 3.2.1 a: Tập dữ liệu insurance.csv*

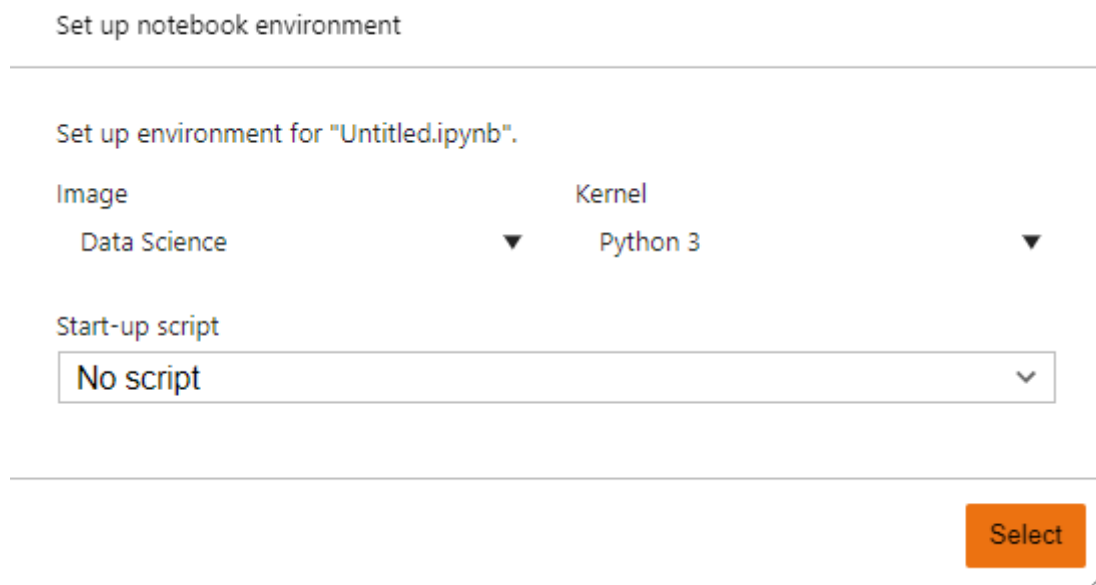
Để tạo một file Notebook ta thực hiện:

Vào File → New → Notebook



*Hình 3.2.1 b: Tạo file notebook*

Sau đó thực hiện setup môi trường notebook bằng cách chọn Image là Data Science và Kernel là Python 3.



Set up notebook environment

---

Set up environment for "Untitled.ipynb".

Image                      Kernel

Data Science                      Python 3

Start-up script

No script

Select

*Hình 3.2.1 c: Thiết lập môi trường notebook*

### 3.2.2 Xử lý và làm sạch dữ liệu

Đầu tiên tiến hành thêm các thư viện được yêu cầu để làm việc với Data Science và dùng thư viện pandas để đọc tập tin csv.



```
[3]: import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt

insurance_df = pd.read_csv('insurance.csv')

insurance_df
```

```
[3]:
```

	age	sex	bmi	children	smoker	region	charges
0	19	female	27.900	0	yes	southwest	16884.92400
1	18	male	33.770	1	no	southeast	1725.55230
2	28	male	33.000	3	no	southeast	4449.46200
3	33	male	22.705	0	no	northwest	21984.47061
4	32	male	28.880	0	no	northwest	3866.85520
...	...	...	...	...	...	...	...
1333	50	male	30.970	3	no	northwest	10600.54830
1334	18	female	31.920	0	no	northeast	2205.98080
1335	18	female	36.850	0	no	southeast	1629.83350
1336	21	female	25.800	0	no	southwest	2007.94500
1337	61	female	29.070	0	yes	northwest	29141.36030

### Hình 3.2.2 a: Đọc tập tin insurance.csv

Tiếp theo thực hiện xem một số thông tin tổng quan như là số lượng, giá trị trung bình, giá trị lớn nhất, nhỏ nhất của từng cột thuộc tính của tập dữ liệu.

```
[10]: #Xem một số thông tin về tập dữ liệu

insurance_df.describe()
```

```
[10]:
```

	age	bmi	children	charges
count	1338.000000	1338.000000	1338.000000	1338.000000
mean	39.207025	30.663397	1.094918	13270.422265
std	14.049960	6.098187	1.205493	12110.011237
min	18.000000	15.960000	0.000000	1121.873900
25%	27.000000	26.296250	0.000000	4740.287150
50%	39.000000	30.400000	1.000000	9382.033000
75%	51.000000	34.693750	2.000000	16639.912515
max	64.000000	53.130000	5.000000	63770.428010

### Hình 3.2.2b: Một số thông tin về dữ liệu

Dùng hàm kiểm tra null để xem tập dữ liệu có chứa giá trị null hay không.

```
[6]: #Kiểm tra có giá trị NULL trong tập dữ liệu hay không?
insurance_df.isnull().sum()

[6]: age          0
sex            0
bmi           0
children      0
smoker        0
region        0
charges       0
dtype: int64
```

### Hình 3.2.2 c: Kết quả kiểm tra giá trị null của tập dữ liệu

Do tập dữ liệu vẫn có các giá trị là chữ nên để trực quan dữ liệu và đào tạo được mô hình máy học thì dữ liệu cần phải được chuyển sang dữ liệu số.

Để thực hiện việc này, đầu tiên ta dùng phương thức unique() để biết được các giá trị khác nhau trong cột thuộc tính, sau đó dùng phương thức apply lambda để chuyển đổi dữ liệu.

```
[11]: # Chuyển đổi dữ liệu sang dữ liệu số
insurance_df['sex'].unique()

insurance_df['sex'] = insurance_df['sex'].apply(lambda x: 0 if x == 'female' else 1)

insurance_df.head()
```

```
[11]:   age  sex  bmi  children  smoker  region  charges
0   19    0  27.900         0     yes southwest  16884.92400
1   18    1  33.770         1      no  southeast   1725.55230
2   28    1  33.000         3      no  southeast   4449.46200
3   33    1  22.705         0      no  northwest  21984.47061
4   32    1  28.880         0      no  northwest   3866.85520
```

### Hình 3.2.2 d: Kết quả chuyển đổi dữ liệu ở cột sex của tập dữ liệu

Tiến hành thực hiện tương tự như vậy để chuyển đổi dữ liệu ở cột smoker.

```
[12]: insurance_df['smoker'].unique()

insurance_df['smoker'] = insurance_df['smoker'].apply(lambda x: 0 if x == 'no' else 1)

insurance_df.head()
```

```
[12]:
```

	age	sex	bmi	children	smoker	region	charges
0	19	0	27.900	0	1	southwest	16884.92400
1	18	1	33.770	1	0	southeast	1725.55230
2	28	1	33.000	3	0	southeast	4449.46200
3	33	1	22.705	0	0	northwest	21984.47061
4	32	1	28.880	0	0	northwest	3866.85520

Hình 3.2.2 e: Kết quả chuyển đổi dữ liệu ở cột smoker

Tương tự đối với cột region do có bốn thuộc tính nên ta sẽ tiến hành tách thành ba cột, nếu dòng đó thuộc region nào thì region đó mang giá trị 1 còn các region còn lại mang giá trị 0.

```
[13]: insurance_df['region'].unique()

[13]: array(['southwest', 'southeast', 'northwest', 'northeast'], dtype=object)

[14]: region_dummies = pd.get_dummies(insurance_df['region'], drop_first = True)

region_dummies
```

```
[15]: insurance_df = pd.concat([insurance_df, region_dummies], axis = 1)

insurance_df.head()
```

```
[15]:
```

	age	sex	bmi	children	smoker	region	charges	northwest	southeast	southwest
0	19	0	27.900	0	1	southwest	16884.92400	0	0	1
1	18	1	33.770	1	0	southeast	1725.55230	0	1	0
2	28	1	33.000	3	0	southeast	4449.46200	0	1	0
3	33	1	22.705	0	0	northwest	21984.47061	1	0	0
4	32	1	28.880	0	0	northwest	3866.85520	1	0	0

Hình 3.2.2 g: Kết quả sau khi chuyển đổi cột region

Cuối cùng là dùng lệnh drop loại bỏ cột region khỏi tập dữ liệu:

```
[16]: insurance_df.drop(['region'], axis = 1, inplace = True)
```

```
insurance_df
```

```
[16]:
```

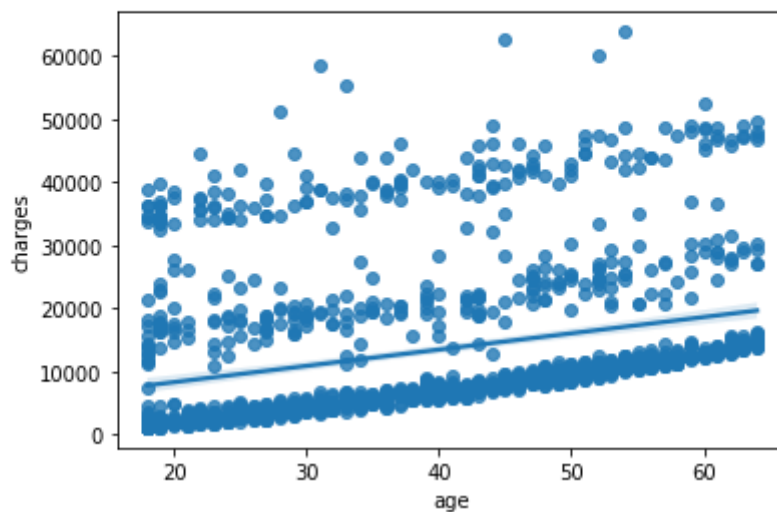
	age	sex	bmi	children	smoker	charges	northwest	southeast	southwest
0	19	0	27.900	0	1	16884.92400	0	0	1
1	18	1	33.770	1	0	1725.55230	0	1	0
2	28	1	33.000	3	0	4449.46200	0	1	0
3	33	1	22.705	0	0	21984.47061	1	0	0
4	32	1	28.880	0	0	3866.85520	1	0	0
...	...	...	...	...	...	...	...	...	...
1333	50	1	30.970	3	0	10600.54830	1	0	0
1334	18	0	31.920	0	0	2205.98080	0	0	0
1335	18	0	36.850	0	0	1629.83350	0	1	0
1336	21	0	25.800	0	0	2007.94500	0	0	1
1337	61	0	29.070	0	1	29141.36030	1	0	0

1338 rows × 9 columns

Hình 3.2.2 h: Kết quả sau khi xử lý tập dữ liệu

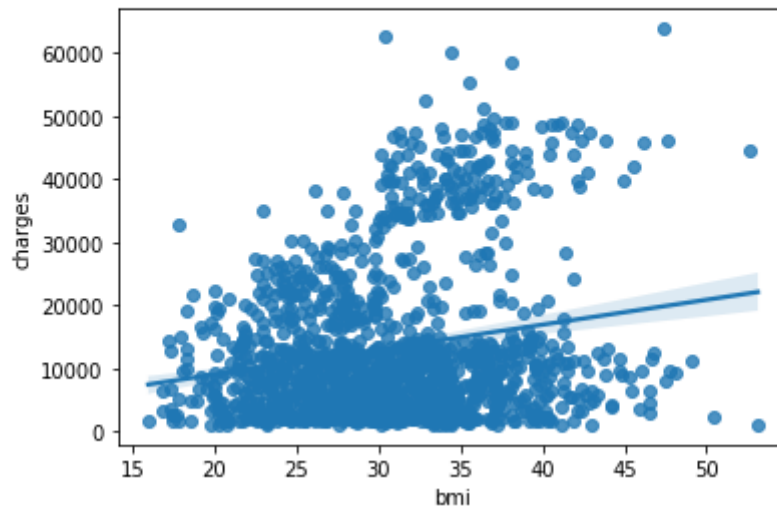
Tiếp theo tiến hành trực quan dữ liệu dưới dạng biểu đồ bằng regplot để thấy được mối quan hệ giữa các cột đầu vào và giá trị đầu ra.

```
[24]: sns.regplot(x='age', y='charges', data=insurance_df)
plt.show()
```



Hình i: Mối quan hệ giữa age và charge

```
[25]: sns.regplot(x='bmi', y='charges', data=insurance_df)
plt.show()
```



*Hình j: Mối quan hệ giữa bmi và charge*

Với các kết quả trên có thể nhận thấy rằng các mối quan hệ này là mối quan hệ tuyến tính, tức là khi giá trị trục ngang tăng thì giá trị trục đứng cũng tăng theo.

### 3.2.3 Phân tách tập dữ liệu

Trước tiên ta gọi X là các giá trị đầu vào, y là giá trị đầu ra chi phí, tức là với tập dữ liệu hiện tại ta sẽ loại bỏ cột charge ban đầu và thêm vào một cột charge rỗng mới.

```
[27]: # Gọi X là dữ liệu đầu vào, y là dữ liệu đầu ra

X = insurance_df.drop(columns = ['charges'])
y = insurance_df['charges']
```

*Hình 3.2.3 a: Loại bỏ cột charge đầu vào và thêm vào cột charge đầu ra*

Thực hiện chuyển đổi X, y sang kiểu dữ liệu float32 và định dạng kết quả đầu ra.

```
[33]: X = np.array(X).astype('float32')
y = np.array(y).astype('float32')
```

```
[34]: y = y.reshape(-1,1)

y.shape
```

### Hình 3.2.3 b : Chuyển đổi dữ liệu X, y

Tiến hành phân tách tập dữ liệu làm hai phần là tập đào tạo và tập kiểm tra với tỉ lệ 0.7 : 0.3 bằng thư viện sklearn.

```
[39]: from sklearn.model_selection import train_test_split
      X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.3)

[40]: X_train.shape

[40]: (936, 8)

[41]: X_test.shape

[41]: (402, 8)
```

### Hình 3.2.3 c: Kết quả phân tách tập dữ liệu

Do dữ liệu có 1338 dòng nên sau khi tách phần đào tạo sẽ là 70% của 1338 là 936 dòng và tập kiểm tra là 402 dòng.

Ta có thể dùng hàm LinearRegression của thư viện sklearn để thực hiện in ra phần trăm độ chính xác của mô hình phân tách trên.

```
[42]: #Train
      from sklearn.linear_model import LinearRegression
      from sklearn.metrics import mean_squared_error, accuracy_score

      regresssion_model_sklearn = LinearRegression()
      regresssion_model_sklearn.fit(X_train, y_train)

      regresssion_model_sklearn_accuracy = regresssion_model_sklearn.score(X_test, y_test)
      regresssion_model_sklearn_accuracy

[42]: 0.7585295192349687
```

### Hình 3.2.3 d: Kết quả phần trăm độ chính xác của mô hình

## 3.2.4 Đào tạo mô hình dùng SageMaker

Trong phần này chúng ta sẽ cần sử dụng đến Amazon S3 để lưu trữ dữ liệu. Chúng ta sẽ sử dụng Bucket mặc định của AWS để làm nơi lưu trữ dữ liệu đào tạo và dữ liệu đầu ra. Chúng ta cũng cần cung cấp vai trò IAM cho SageMaker để thực hiện các nhiệm vụ thay mặt chúng ta.

Đầu tiên ta cần phải thêm thư viện chính của đề tài này là sagemaker, ngoài ra cần thêm thư viện boto3 – một AWS Python SDK cho phép các nhà phát triển viết ứng dụng sử dụng các dịch vụ như Amazon S3 hoặc EC2.

```
[44]: import sagemaker
import boto3

# Tạo một phiên sagemaker
sagemaker_session = sagemaker.Session()

# Định nghĩa S3 bucket và thư mục con prefix
bucket = 'sagemaker-studio-727181768726-subniu90fo'
prefix = 'linear-model'

#Xác định IAM role
role = sagemaker.get_execution_role()
print(role)

arn:aws:iam::727181768726:role/LabRole
```

*Hình 3.2.4 a: Kết quả vai trò IAM của SageMaker*

Tiếp theo thực hiện chuyển đổi  $y_{\text{train}}$  và  $y_{\text{test}}$  đã định nghĩa là giá trị đầu ra bên trên sang dạng vector.

```
[49]: #Chuyển y_train và y_test sang dạng vector

y_train = y_train[:,0]
y_test = y_test[:,0]

[50]: y_train.shape

[50]: (936,)
```

```
[51]: y_test.shape

[51]: (402,)
```

*Hình 3.2.4 b: Chuyển đổi sang dạng vector*

Thực hiện chuyển đổi dữ liệu đào tạo thành định dạng đầu vào tương thích với Sagemaker và xóa bộ nhớ đệm trong bộ nhớ.

```
[52]: # chuyển đổi dữ liệu training thành định dạng đầu vào tương thích với Sagemaker (RecordIO)

import io

import sagemaker.amazon.common as smac

buf = io.BytesIO()
smac.write_numpy_to_dense_tensor(buf, X_train, y_train)

buf.seek(0)
```

[52]: 0

### Hình 3.2.4 c: Chuyển đổi dữ liệu đào tạo

Tiến hành tải dữ liệu đào tạo lên S3 bằng cách định nghĩa thêm một key. Vị trí dữ liệu tải lên được định dạng như sau: đầu tiên là bucket, ở đây là bucket mặc định của AWS S3, sau đó là prefix hay thư mục con của bucket, tiếp theo là thư mục train và cuối cùng là key vừa định nghĩa.

```
[53]: # Tải dữ liệu lên S3

import os

key = 'linear-model-data'

# tải dữ liệu ở định dạng record-io lên S3
boto3.resource('s3').Bucket(bucket).Object(os.path.join(prefix, 'train', key)).upload_fileobj(buf)

s3_train_data = 's3://{}/{}/train/{}'.format(bucket, prefix, key)
print('Dữ liệu đã được tải lên tại: {}'.format(s3_train_data))

Dữ liệu đã được tải lên tại: s3://sagemaker-studio-727181768726-subniu90fo/linear-model/train/linear-model-data
```

### Hình 3.2.4 e: Kết quả vị trí nơi lưu trữ dữ liệu đào tạo

Tương tự như trên tiến hành tạo nơi lưu trữ dữ liệu đầu ra.

```
[54]: # Tạo nơi để lưu trữ output

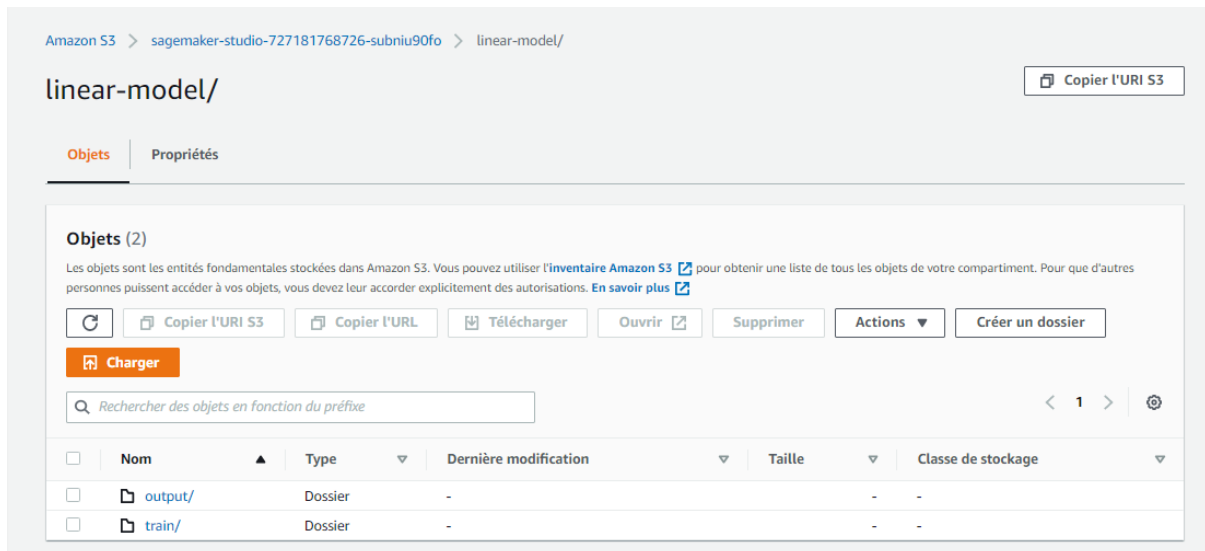
output_location = 's3://{}/{}/output'.format(bucket, prefix)

print("Output được lưu tại vị trí: {}".format(output_location))

Output được lưu tại vị trí: s3://sagemaker-studio-727181768726-subniu90fo/linear-model/output
```

### Hình 3.2.4. g: Kết quả vị trí lưu trữ dữ liệu đầu ra





Hình 3.2.4 h: Kết quả được tạo ra ở S3

Bước tiếp theo thực hiện đào tạo mô hình bằng cách tạo container và chỉ định tên thuật toán triển khai, ở đây dùng thuật toán linear-learner.

```
[55]: # Tạo container và tên thuật toán để train

from sagemaker.amazon.amazon_estimator import get_image_uri

container = get_image_uri(boto3.Session().region_name, 'linear-learner')
```

Hình 3.2.4 i: Chỉ định tên thuật toán để đào tạo

Tiếp theo chuyển container vừa tạo và các loại instances, đường dẫn đầu ra, phiên sagemarket đã định nghĩa từ trước để đào tạo mô hình.

```
[56]: # Chuẩn bị trước khi training

linear = sagemaker.estimator.Estimator(container,
    role,
    train_instance_count = 1,
    train_instance_type = 'ml.c4.xlarge',
    output_path = output_location,
    sagemaker_session = sagemaker_session)
```

Hình 3.2.4 j: Chuẩn bị trước khi đào tạo mô hình

Thiết lập các tham số tùy chỉnh để đào tạo mô hình như là feature\_dim tức là số lượng cột đầu vào (ở đây là 8), tiếp theo là thể loại dự đoán predictor\_type (ở đây là loại hồi quy) và số lượng mô hình đào tạo num\_models.

```
linear.set_hyperparameters(feature_dim = 8,
                           predictor_type = 'regressor',
                           mini_batch_size = 100,
                           epochs = 100,
                           num_models = 32,
                           loss = 'absolute_loss')
```

Hình 3.2.4 k: Các tham số đào tạo mô hình

Do sử dụng môi trường phát triển tích hợp của AWS nên việc đào tạo mô hình được thực hiện đơn giản chỉ bằng một dòng lệnh.

```
# Training mô hình với dữ liệu từ S3

linear.fit({'train': s3_train_data})
```

```
ost": "algo-1", "Operation": "training", "epoch": 10, "model": 23}, "Metrics": {"train_absolute_loss_objective": {"sum":
0.36818597581651474, "count": 1, "min": 0.36818597581651474, "max": 0.36818597581651474}}}
#metrics {"StartTime": 1637416475.5632389, "EndTime": 1637416475.5632536, "Dimensions": {"Algorithm": "Linear Learner", "Ho
st": "algo-1", "Operation": "training", "epoch": 10, "model": 24}, "Metrics": {"train_absolute_loss_objective": {"sum":
0.6856172392103407, "count": 1, "min": 0.6856172392103407, "max": 0.6856172392103407}}}
#metrics {"StartTime": 1637416475.5633042, "EndTime": 1637416475.5633206, "Dimensions": {"Algorithm": "Linear Learner", "Ho
st": "algo-1", "Operation": "training", "epoch": 10, "model": 25}, "Metrics": {"train_absolute_loss_objective": {"sum":
0.6856088807847764, "count": 1, "min": 0.6856088807847764, "max": 0.6856088807847764}}}
#metrics {"StartTime": 1637416475.563369, "EndTime": 1637416475.5633845, "Dimensions": {"Algorithm": "Linear Learner", "Ho
st": "algo-1", "Operation": "training", "epoch": 10, "model": 26}, "Metrics": {"train_absolute_loss_objective": {"sum": 0.
6853890185885959, "count": 1, "min": 0.6853890185885959, "max": 0.6853890185885959}}}
#metrics {"StartTime": 1637416475.563433, "EndTime": 1637416475.5634491, "Dimensions": {"Algorithm": "Linear Learner", "Ho
st": "algo-1", "Operation": "training", "epoch": 10, "model": 27}, "Metrics": {"train_absolute_loss_objective": {"sum": 0.
6856346342298719, "count": 1, "min": 0.6856346342298719, "max": 0.6856346342298719}}}
#metrics {"StartTime": 1637416475.563498, "EndTime": 1637416475.5635145, "Dimensions": {"Algorithm": "Linear Learner", "Ho
st": "algo-1", "Operation": "training", "epoch": 10, "model": 28}, "Metrics": {"train_absolute_loss_objective": {"sum": 0.
6900124104817709, "count": 1, "min": 0.6900124104817709, "max": 0.6900124104817709}}}
#metrics {"StartTime": 1637416475.5635622, "EndTime": 1637416475.5635772, "Dimensions": {"Algorithm": "Linear Learner", "Ho
st": "algo-1", "Operation": "training", "epoch": 10, "model": 29}, "Metrics": {"train_absolute_loss_objective": {"sum":
0.6891133456759982, "count": 1, "min": 0.6891133456759982, "max": 0.6891133456759982}}}
#metrics {"StartTime": 1637416475.5636256, "EndTime": 1637416475.563641, "Dimensions": {"Algorithm": "Linear Learner", "Ho
st": "algo-1", "Operation": "training", "epoch": 10, "model": 30}, "Metrics": {"train_absolute_loss_objective": {"sum": 0.
6976962746514215, "count": 1, "min": 0.6976962746514215, "max": 0.6976962746514215}}}
#metrics {"StartTime": 1637416475.5636897, "EndTime": 1637416475.5637054, "Dimensions": {"Algorithm": "Linear Learner", "Ho
st": "algo-1", "Operation": "training", "epoch": 10, "model": 31}, "Metrics": {"train_absolute_loss_objective": {"sum":
0.6912968529595269, "count": 1, "min": 0.6912968529595269, "max": 0.6912968529595269}}}
```

Hình 3.2.4 m: Kết quả sau khi đào tạo

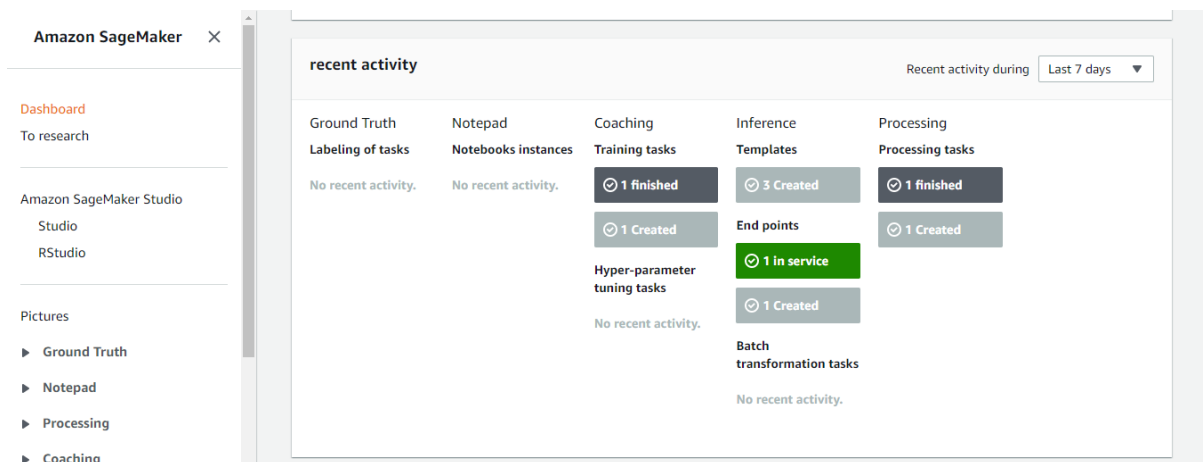
### 3.2.5 Triển khai mô hình

Bước triển khai này cũng được được Amazon SageMaker tích hợp để phát triển nên việc triển khai cũng diễn ra rất đơn giản.

```
[73]: linear_regressor = linear.deploy(initial_instance_count = 1, instance_type = 'ml.m4.xlarge')
-----!
```

Hình 3.2.5 a: Kết quả khi triển khai mô hình

Khi chúng ta triển khai mô hình dùng SageMaker thì hệ thống sẽ tạo ra một điểm truy cập end point để người dùng có thể truy cập.



*Hình 3.2.5 b: Điểm truy cập end point*

### 3.2.6 Kiểm tra

Để kiểm tra mô hình vừa triển khai trước tiên cần chuyển dữ liệu sang dạng text/csv và dùng hàm dự đoán predict để in ra kết quả dự đoán.

```
[59]: # Kiểm tra

from sagemaker.predictor import csv_serializer, json_deserializer

linear_regressor.serializer = csv_serializer
linear_regressor.deserializer = json_deserializer

[60]: result = linear_regressor.predict(X_test)

result

[60]: {'predictions': [{'score': 11428.431640625},
  {'score': 10293.044921875},
  {'score': 13296.986328125},
  {'score': 11989.4453125},
  {'score': 33655.54296875},
  {'score': 35843.34375},
  {'score': 40674.7734375},
  {'score': 14787.56640625},
  {'score': 12711.39453125},
  {'score': 41572.49609375},
  {'score': 3471.986328125},
  {'score': 9735.314453125},
  {'score': 43267.234375},
  {'score': 37217.76953125},
```

*Hình 3.2.6 a : Kết quả dự đoán*

Ta cũng có thể chuyển kết quả sang dạng array bằng cách chuyển đổi.

```
[61]: predictions = np.array([r['score'] for r in result['predictions']])

predictions

[61]: array([11428.43164062, 10293.04492188, 13296.98632812, 11989.4453125 ,
          33655.54296875, 35843.34375 , 40674.7734375 , 14787.56640625,
          12711.39453125, 41572.49609375, 3471.98632812, 9735.31445312,
          43267.234375 , 37217.76953125, 9815.46875 , 31516.3984375 ,
          37513.1015625 , 2032.50683594, 12553.49609375, 6793.51464844,
          36357.65625 , 37902.546875 , 29417.046875 , 10285.75488281,
           8002.60546875, 9370.86230469, 9544.15039062, 5865.45898438,
          36784.421875 , 2750.80517578, 8343.77539062, 160.06835938,
          4253.82421875, 9018.109375 , 4123.26464844, 10277.37109375,
          5129.21582031, 11138.86035156, 32508.80859375, 2128.74951172,
          4225.00195312, 4168.2890625 , 9024.61425781, 5785.03320312,
          15999.55664062, 13745.046875 , 36750.74609375, 5103.0546875 ,
          41335.1015625 , 10384.00976562, 37607.3359375 , 8490.109375 ,
          2114.88476562, 1676.48339844, 11178.37304688, 7210.87109375,
          5519.90820312, 2647.28759766, 33431.07421875, 39670.2734375 ,
          6481.37695312, 7020.45507812, 6983.38183594, 12889.74023438,
          10697.0078125 , 31140.78515625, 4249.1015625 , 7687.29492188,
          41892.09765625, 9287.34765625, 12876.06445312, 12340.9375 ,
          9927.57421875, 6335.13378906, 13052.6171875 , 10839.83984375,
```

Hình 3.2.6 b: Kết quả dưới dạng array

Tiếp theo thực hiện in kết quả thực tế và kết quả dự đoán để so sánh.

```
[62]: # Kiểm tra kết quả

df = pd.DataFrame({'Thực tế': y_test.flatten(), 'Dự đoán': predictions.flatten()})

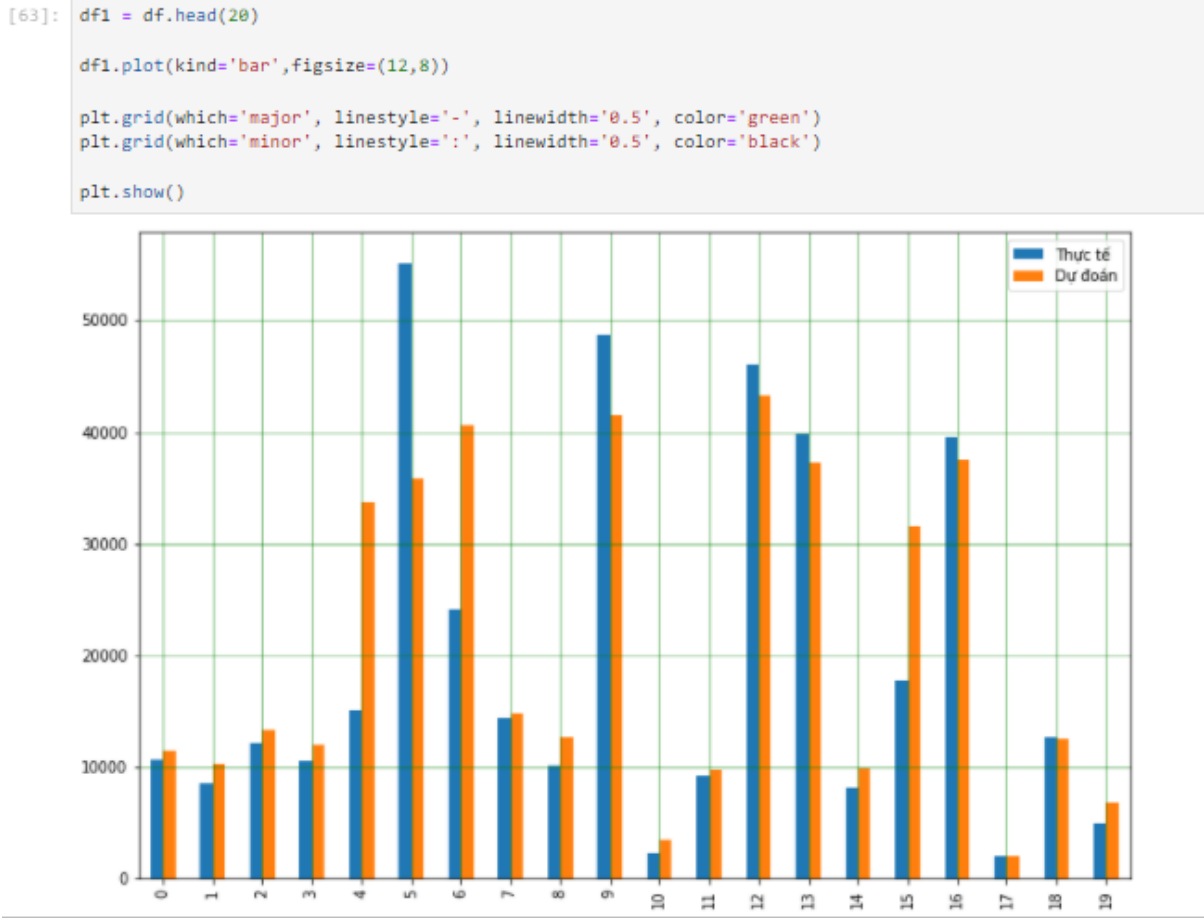
df
```

```
[62]:
```

	Thực tế	Dự đoán
0	10601.412109	11428.431641
1	8547.691406	10293.044922
2	12105.320312	13296.986328
3	10564.884766	11989.445312
4	15006.579102	33655.542969
...	...	...
397	10965.446289	10555.230469
398	5438.749023	7752.345703
399	12557.605469	12258.828125
400	3875.734131	5845.537109
401	33750.292969	30564.644531

402 rows × 2 columns

Hình 3.2.6 c: Kết quả thực tế và kết quả dự đoán



Hình 3.2.6 d: Kết quả dưới dạng biểu đồ

Như vậy với độ chính xác dự đoán là khoảng 75% thì mô hình đã dự đoán được chi phí bảo hiểm của khách hàng khá chính xác so với kết quả thực tế.

Cuối cùng xóa end point để không bị AWS tính phí.

```
[71]: # Xóa end-point

linear_regressor.delete_endpoint()
```

Hình 3.2.6 e: Xóa điểm cuối end point

## CHƯƠNG 4 TỔNG KẾT

### 4.1 Kết quả đạt được:

Trong quá trình thực hiện đề tài, nhóm em đã đạt được những thành quả sau:

- Biết được các dịch vụ được cung cấp bởi AWS, cụ thể là Amazon SageMaker, Amazon S3.
  - Các bước để thực hiện một dự án Machine Learning dùng Amazon SageMaker.
  - Biết được một phần nhỏ của lĩnh vực Data Science.
  - Áp dụng được các dịch vụ trên nền tảng đám mây của AWS cung cấp để thực hiện đề tài.
- ⇒ Những điều rút ra được sau khi làm đề tài: Amazon SageMaker sẽ khiến việc xây dựng các mô hình ML trở nên dễ dàng bằng cách cung cấp mọi thứ chúng ta cần để nhanh chóng kết nối với dữ liệu đào tạo, cũng như chọn thuật toán và framework tốt nhất cho ứng dụng của chúng ta. Đồng thời, dịch vụ này cũng quản lý tất cả cơ sở hạ tầng nền tảng để có thể đào tạo mô hình ở quy mô petabyte.

### 4.2 Những hạn chế:

- Ứng dụng minh họa còn cơ bản chưa đến mức nâng cao.
- Độ chính xác của việc dự đoán chưa cao lắm chỉ khoảng 75%.
- Thuật toán thực hiện còn ở mức cơ bản.

### 4.3 Hướng phát triển:

- Cải tiến thuật toán để mô hình có thể đưa ra dự đoán chính xác hơn nữa.
- Dùng AWS Lambda để tạo ra các điểm truy cập cho người dùng.
- Cải tiến mã code để tối ưu hơn.

## TÀI LIỆU THAM KHẢO

1. Kaggle: <https://www.kaggle.com/mirichoi0218/insurance>
2. Wikipedia: [https://vi.wikipedia.org/wiki/Amazon\\_Web\\_Services](https://vi.wikipedia.org/wiki/Amazon_Web_Services)  
[https://vi.wikipedia.org/wiki/H%E1%BB%8Dc\\_m%C3%A1y](https://vi.wikipedia.org/wiki/H%E1%BB%8Dc_m%C3%A1y)
3. Bizfly Cloud: <https://bizflycloud.vn/tin-tuc/mot-so-vi-du-noi-bat-ve-dien-toan-dam-may-cloud-computing-20180611015908902.htm>
4. CUONG QUACH: <https://cuongquach.com/amazon-s3-la-gi.html>
5. ICHI.PRO <https://ichi.pro/vi/amazon-sagemaker-cho-hoc-sau-may-hoc-huong-dan-toan-dien-178052897125174>
6. AWS: <https://aws.amazon.com/vi/sagemaker/>
7. Towards data science: <https://towardsdatascience.com/machine-learning-on-aws-sagemaker-b23e32503106>