**MỤC LỤC**

Contents

[CHƯƠNG 1: GIỚI THIỆU TỔNG QUÁT BÀI TOÁN 4](#_Toc187247214)

[1.1 Lý do chọn đề tài 4](#_Toc187247215)

[1.2 Mục tiêu nghiên cứu 5](#_Toc187247216)

[1.2.1 Mục tiêu tổng quát 5](#_Toc187247217)

[1.2.2 Mục tiêu cụ thể 5](#_Toc187247218)

[1.3 Đối tượng nghiên cứu và phạm vi nghiên cứu 6](#_Toc187247219)

[1.3.1 Đối tượng nghiên cứu 6](#_Toc187247220)

[1.3.2 Phạm vi nghiên cứu 6](#_Toc187247221)

[CHƯƠNG 2: CÁC KỸ THUẬT SỬ DỤNG TRONG BÀI TOÁN 7](#_Toc187247222)

[2.1 Tổng quan về học máy 7](#_Toc187247223)

[2.1.1 Khái niệm học máy 7](#_Toc187247224)

[2.1.2 Phân loại nhóm thuật toán trong học máy 7](#_Toc187247225)

[2.1.3 Các mô hình học máy áp dụng 9](#_Toc187247226)

[2.2. Các phương pháp đánh giá mô hình 18](#_Toc187247227)

[CHƯƠNG 3: ỨNG DỤNG PHƯƠNG PHÁP VÀ XÂY DỰNG MÔ HÌNH 20](#_Toc187247228)

[3.1 Công cụ sử dụng và quy trình thu thập dữ liệu 20](#_Toc187247229)

[3.1.1 Công cụ sử dụng 20](#_Toc187247230)

[3.1.2 Quy trình thu thập dữ liệu 20](#_Toc187247231)

[3.1.3 Kết quả đạt được 25](#_Toc187247232)

[3.3 Tiền xử lý dữ liệu 27](#_Toc187247233)

[3.3.1 Xử lý giá trị dữ liệu 27](#_Toc187247234)

[3.3.2 Chuẩn hóa cấu trúc 27](#_Toc187247235)

[3.4 Phân tích dữ liệu 28](#_Toc187247236)

[3.5 Xây dựng và huấn luyện mô hình 32](#_Toc187247237)

[CHƯƠNG 4: KẾT QUẢ VÀ ĐÁNH GIÁ MÔ HÌNH 36](#_Toc187247238)

[4.1 Kết quả 36](#_Toc187247239)

[4.2 Đánh giá mô hình 38](#_Toc187247240)

[4.3 Xây dựng giao diện ứng dụng 41](#_Toc187247241)

[KẾT LUẬN 43](#_Toc187247242)

[TÀI LIỆU THAM KHẢO 43](#_Toc187247243)

# CHƯƠNG 1: GIỚI THIỆU TỔNG QUÁT BÀI TOÁN

## Lý do chọn đề tài

Dự đoán giá vàng là một trong những thách thức quan trọng trong lĩnh vực tài chính, đặc biệt là trong bối cảnh thị trường biến động liên tục. Giá vàng không chỉ ảnh hưởng đến quyết định đầu tư của các nhà đầu tư mà còn gắn liền với sự ổn định của nền kinh tế toàn cầu. Tuy nhiên, việc dự đoán giá vàng là một bài toán phức tạp và đa chiều, do ảnh hưởng của nhiều yếu tố như tình hình kinh tế, chính trị, và tâm lý thị trường.

Trong quá trình dự đoán giá vàng, các phương pháp thống kê truyền thống thường gặp hạn chế trong việc mô hình hóa các mối quan hệ phức tạp giữa các yếu tố ảnh hưởng và giá cả. Sự biến động nhanh chóng của thị trường và sự phức tạp của dữ liệu đòi hỏi sự linh hoạt và mạnh mẽ hơn từ các phương pháp dự đoán.

Trong những năm gần đây, sự phát triển mạnh mẽ của trí tuệ nhân tạo, đặc biệt là các kỹ thuật học máy, đã mở ra cánh cửa cho các phương pháp dự đoán tiên tiến hơn trong việc dự đoán giá vàng. Các thuật toán học máy có khả năng học từ dữ liệu lớn và phức tạp có thể phát hiện ra các mẫu và xu hướng ẩn sau những biến động của thị trường, từ đó cung cấp các dự báo về giá vàng với độ chính xác cao hơn.

Việc áp dụng các kỹ thuật học máy vào dự đoán giá vàng không chỉ mang lại lợi ích cho các nhà đầu tư trong việc ra quyết định đầu tư thông minh mà còn hỗ trợ cho các tổ chức tài chính và chính phủ trong việc quản lý rủi ro và phát triển kinh tế. Đồng thời, nó cũng tạo ra cơ hội mới để tối ưu hóa các chiến lược giao dịch và tăng cường hiệu suất trong thị trường vàng.

Từ những thách thức và tiềm năng của việc dự đoán giá vàng, việc lựa chọn đề tài "Dự đoán giá vàng sử dụng các kỹ thuật học máy" là một bước đi mang tính chiến lược, nhằm khai thác và áp dụng sức mạnh của trí tuệ nhân tạo vào lĩnh vực tài chính, góp phần vào sự phát triển bền vững của thị trường vàng toàn cầu.

## 1.2 Mục tiêu nghiên cứu

### 1.2.1 Mục tiêu tổng quát

Xây dựng mô hình học máy hiệu quả để dự đoán giá vàng trong tương lai với độ chính xác cao

### 1.2.2 Mục tiêu cụ thể

* Đối với việc thu thập dữ liệu cho đề tài dự đoán giá vàng sử dụng các kỹ thuật học máy, có thể lựa chọn các nguồn dữ liệu uy tín như các trang web chuyên về thị trường vàng, các sàn giao dịch vàng lớn, và các tổ chức tài chính có cung cấp thông tin về giá vàng.
* Dữ liệu thu thập được có thể bao gồm các thông tin như giá vàng trong thời gian trước đó, các chỉ số kinh tế có liên quan như tỷ giá hối đoái, giá dầu, chỉ số chứng khoán, và các thông tin thị trường quốc tế.
* Sau khi thu thập dữ liệu, việc tiền xử lý là bước quan trọng để loại bỏ nhiễu, xử lý dữ liệu thiếu và chuẩn hóa dữ liệu. Việc này giúp tạo ra bộ dữ liệu sạch sẽ và đồng nhất để có thể áp dụng vào các mô hình học máy.
* Phân tích dữ liệu là bước tiếp theo để phân tích và tìm ra các đặc trưng quan trọng và mối quan hệ giữa các biến trong dữ liệu. Điều này giúp cải thiện hiểu biết về dữ liệu và xây dựng các mô hình dự đoán chính xác hơn.
* Cuối cùng, việc xây dựng mô hình học máy sẽ giúp dự đoán giá vàng trong tương lai. Việc đánh giá và so sánh hiệu suất của các mô hình giúp chọn ra mô hình tốt nhất cho mục đích dự đoán giá vàng.

## 1.3 Đối tượng nghiên cứu và phạm vi nghiên cứu

### 1.3.1 Đối tượng nghiên cứu

* Dữ liệu về giá vàng: Đối tượng nghiên cứu chính là dữ liệu về giá vàng, bao gồm các biến số như giá mở cửa, giá cao nhất, giá thấp nhất, giá đóng cửa và khối lượng giao dịch. Thông tin này được thu thập từ các nguồn uy tín như các sàn giao dịch vàng và các trang web chuyên về thị trường vàng.
* Mô hình học máy: Trong quá trình nghiên cứu, một số mô hình như AdaBoost, XGBoost, LightGBM cũng là đối tượng nghiên cứu quan trọng. Việc xây dựng và tối ưu hóa các mô hình này để dự đoán giá vàng là mục tiêu của đề tài.

### 1.3.2 Phạm vi nghiên cứu

Đề tài tập trung nghiên cứu áp dụng một số mô hình học máy AdaBoost, XGBoost, LightGBM để dự đoán giá đóng của giá vàng.

# CHƯƠNG 2: CÁC KỸ THUẬT SỬ DỤNG TRONG BÀI TOÁN

## 2.1 Tổng quan về học máy

### 2.1.1 Khái niệm học máy

Học máy hay còn được gọi là máy học là một lĩnh vực của trí tuệ nhân tạo liên quan đến việc nghiên cứu và xây dựng các kỹ thuật cho phép các chương trình máy tính “học” về cách hoàn thành các nhiệm vụ cụ thể và cách cải thiện hiệu suất theo thời gian. Các mô hình học máy yêu cầu dữ liệu để huấn luyện và đánh giá mô hình.

### 2.1.2 Phân loại nhóm thuật toán trong học máy

Các kỹ thuật học máy chủ yếu thuộc 2 nhóm chính: Học có giám sát (*Supervised Learning*) và Học không giám sát (*Unsupervised Learning*). Ngoài ra còn có Học tăng cường (*Reinforcement Learning*)[1].

2.1.2.1 Học có giám sát (*Supervised Learning*)

Trong Học có giám sát, máy tính học cách mô hình hóa các mối quan hệ dựa trên dữ liệu được gán nhãn. Sau khi tìm hiểu cách tốt nhất để mô hình hóa các mối quan hệ cho dữ liệu được gắn nhãn, các thuật toán được huấn luyện được sử dụng cho các bộ dữ liệu mới.

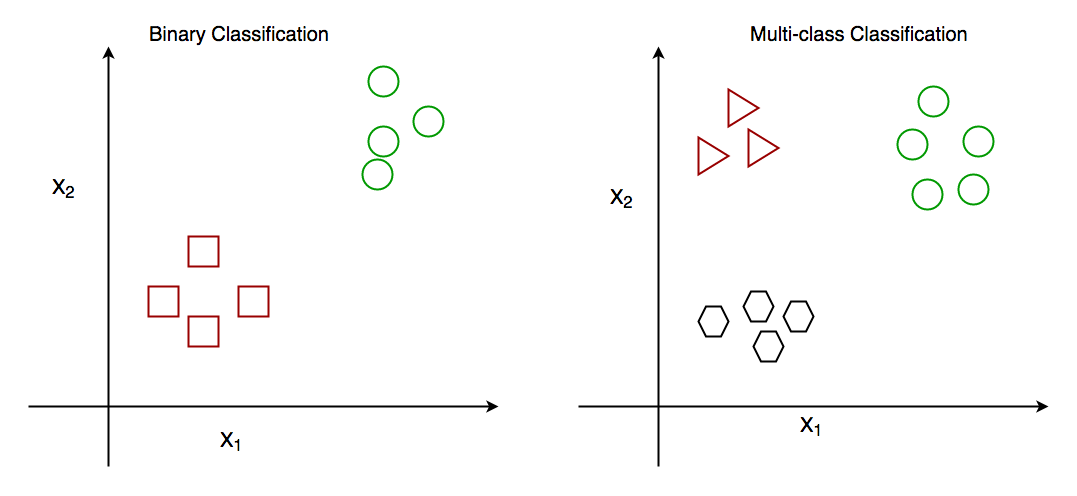
Ưu điểm của Học có giám sát là độ chính xác cao do được huấn luyện trên bộ dữ liệu đã được gán nhãn. Phương pháp này rất phổ biến và hiệu quả trong nhiều ứng dụng dự báo, phân lớp dữ liệu thực tế.

Classification (Phân lớp) và Regression (Hồi quy) là 2 bài toán phổ biến trong học có giám sát.

**Classification (Phân lớp)**

Phân lớp là quá trình tìm kiếm hoặc khám phá một mô hình hoặc chức năng giúp tách dữ liệu thành nhiều lớp phân loại, tức là các giá trị riêng biệt. Trong bài toán phân lớp, dữ liệu được phân loại theo các nhãn khác nhau theo một số tham số được đưa ra trong dữ liệu đầu vào và sau đó các nhãn được dự đoán.

Trong bài toán phân lớp, ta có nhiệm vụ phải dự đoán các biến mục tiêu riêng biệt bằng cách sử dụng các thuộc tính. Ta phải tìm ra ranh giới quyết định có thể phân tách các lớp khác nhau.



*Hình 2.1: Minh họa bài toán phân lớp*

Ví dụ về bài toán phân lớp như: Phân loại email có phải spam hay không, nhận dạng chữ viết tay, ta có ảnh của hàng nghìn ví dụ của mỗi chữ số được viết bởi nhiều người khác nhau. Chúng ta đưa các bức ảnh này vào trong một thuật toán và chỉ cho nó biết mỗi bức ảnh tương ứng với chữ số nào.

**Regression (Hồi quy)**

Hồi quy là quá trình tìm mô hình hoặc hàm để phân biệt dữ liệu thành các giá trị thực liên tục thay vì sử dụng các lớp hoặc giá trị rời rạc. Nó cũng có thể xác định chuyển động phân phối tùy thuộc vào dữ liệu lịch sử. Bởi vì mô hình dự đoán hồi quy dự đoán một đại lượng, do đó, kỹ năng của mô hình phải được báo cáo là lỗi trong các dự đoán đó.

Mục tiêu của bài toán hồi quy là dự đoán một giá trị của một biến phụ thuộc vào giá trị của các thuộc tính độc lập cho trước. Chẳng hạn dự đoán mức thu nhập của một người dựa trên học vấn, quê quán, độ tuổi và công việc, dự đoán giá của một ngôi nhà dựa trên diện tích và vị trí,…

2.1.2.2 Học không giám sát (*Unsupervised Learning)*

Khác với Học có giám sát, chúng ta có thể biết được nhãn của dữ liệu huấn luyện, thì trong Học không giám sát chúng ta không biết được outcome hay nhãn mà chỉ có dữ liệu đầu vào. Thuật toán sẽ dựa vào cấu trúc của dữ liệu để thực hiện một công việc nào đó. Hiểu một cách toán học, Học không giám sát là khi chúng ta chỉ có dữ liệu vào X mà không biết nhãn Y tương ứng.

Clustering (Phân cụm) là một bài toán phổ biến trong Học không giám sát.

Ở bài toán này, Ta có dữ liệu huấn luyện đầu vào X và mục tiêu của bài toán là phân các điểm dữ liệu thành các nhóm nhỏ dựa trên sự liên quan giữa các dữ liệu trong mỗi nhóm. Ví dụ về bài toán này có thể kể đến như: Phân nhóm khách hàng dựa trên thói quen mua sắm, sở thích; Nhóm các văn bản có nội dung tương đồng nhau,…



*Hình 2.2: Minh họa bài toán phân cụm*

### 2.1.3 Các mô hình học máy áp dụng

**AdaBoost (Adaptive Boosting)**

AdaBoost là một thuật toán thuộc họ boosting, được giới thiệu bởi Freund và Schapire vào năm 1997. Nó là một phương pháp ensemble học nhằm cải thiện độ chính xác của các mô hình dự đoán thông qua việc kết hợp nhiều mô hình yếu (weak learners), thường là cây quyết định đơn giản (decision stumps), thành một mô hình mạnh (strong learner).

**Nguyên lý hoạt động**

**Ban đầu**:

* Gán trọng số bằng nhau cho tất cả các mẫu trong tập dữ liệu.
* Các trọng số này đại diện cho tầm quan trọng của từng mẫu dữ liệu.

**Lặp qua nhiều bước**:

* Huấn luyện một mô hình yếu (ví dụ: cây quyết định) trên tập dữ liệu với các trọng số hiện tại.
* Tính lỗi của mô hình yếu trên tập dữ liệu.
* Điều chỉnh trọng số của mẫu:
  + Mẫu được dự đoán sai sẽ được gán trọng số lớn hơn để mô hình tiếp theo tập trung vào chúng.
  + Mẫu được dự đoán đúng sẽ giảm trọng số.
* Tính trọng số (alpha) cho mô hình yếu hiện tại dựa trên độ chính xác của nó.
* Cập nhật tổng hợp dự đoán.

**Kết quả cuối cùng**:

* Mô hình dự đoán mạnh được xây dựng bằng cách kết hợp dự đoán từ tất cả các mô hình yếu, với mỗi mô hình được gán trọng số theo độ chính xác của nó.

**Thuật toán cụ thể**

Giả sử có:

* Tập dữ liệu: (,), (,),…, (, ,) với yi ∈ {−1,1}
* Trọng số: ,​ là trọng số của mẫu i.

Các bước chính:

1. Khởi tạo trọng số:
2. Lặp qua T mô hình yếu:
   * Huấn luyện mô hình yếu ht(x) với trọng số .
   * Tính lỗi ϵt của ​

(Trong đó, II là hàm chỉ báo, bằng 1 khi dự đoán sai.)

* + Tính trọng số của mô hình yếu:
  + Cập nhật trọng số mẫu: wi ← wi ⋅ exp(- ⋅ ⋅
  + Chuẩn hóa trọng số: ← ​

1. Kết hợp dự đoán:
   * Dự đoán cuối cùng: H(x) = sign()

**Ưu điểm**

* Hiệu quả cao: Làm việc tốt với các mô hình yếu như decision stumps.
* Không cần tinh chỉnh nhiều: Chỉ cần chọn mô hình yếu và số vòng lặp T.
* Dễ triển khai: Thuật toán rõ ràng, có nhiều thư viện hỗ trợ.
* Xử lý outliers (một cách giới hạn): Tập trung vào các mẫu khó (sai sót lớn).

**Nhược điểm**

* Nhạy cảm với nhiễu: Các outliers (dữ liệu nhiễu) có thể làm thuật toán tập trung quá mức vào chúng, dẫn đến overfitting.
* Không phù hợp với dữ liệu lớn: AdaBoost có thể chậm khi dữ liệu lớn vì cần huấn luyện nhiều mô hình liên tiếp.
* Hạn chế trong hồi quy: AdaBoost chủ yếu được thiết kế cho phân loại, mặc dù có biến thể AdaBoost.R để áp dụng cho bài toán hồi quy.

**XGBoost (Extreme Gradient Boosting)**

XGBoost là một thuật toán boosting mạnh mẽ được thiết kế để đạt hiệu suất cao cả về tốc độ tính toán lẫn độ chính xác. Nó được phát triển bởi Tianqi Chen và trở nên phổ biến nhờ khả năng chiến thắng trong nhiều cuộc thi học máy. XGBoost là một phiên bản nâng cấp của Gradient Boosting Machine (GBM), tích hợp nhiều tối ưu hóa hiệu suất và khả năng điều chỉnh linh hoạt.

**Nguyên lý hoạt động**

XGBoost thuộc họ gradient boosting, trong đó các mô hình được xây dựng dần dần thông qua việc kết hợp nhiều mô hình yếu (thường là cây quyết định). XGBoost tối ưu hóa bằng cách giảm hàm mất mát thông qua việc cập nhật dựa trên đạo hàm gradient của hàm mục tiêu.

**Quy trình chính:**

* Khởi tạo mô hình: Dự đoán ban đầu là giá trị trung bình (cho hồi quy) hoặc tần suất lớn nhất (cho phân loại).
* Lặp qua các bước boosting:

1. Xây dựng một mô hình cây quyết định dựa trên phần dư (residual) giữa giá trị thực tế và giá trị dự đoán hiện tại.
2. Sử dụng gradient của hàm mất mát để đo mức độ cần cải thiện.
3. Kết hợp mô hình mới vào mô hình hiện tại với một hệ số học (η\etaη, learning rate).

* Tổng hợp dự đoán:

1. Dự đoán cuối cùng là tổng hợp dự đoán của tất cả các cây với trọng số được điều chỉnh.

**Thuật toán cụ thể**

Giả sử tập dữ liệu có n mẫu (xi,yi)

1. Khởi tạo mô hình:
   * Bắt đầu với một giá trị dự đoán ban đầu.
2. Tính gradient và hessian:
   * Với mỗi mẫu, tính gradient (gi) và hessian (hi) dựa trên hàm mất mát L(y, ӯ):
3. **Xây dựng cây quyết định**:

Sử dụng gradient và hessian để tối ưu hóa hàm mục tiêu, tìm điểm chia nhánh tốt nhất tại mỗi bước.

Hàm mục tiêu xấp xỉ tại bước t:

Trong đó:

Ω(ft)= : hàm chính quy hóa để giảm overfitting.

γ: chi phí thêm một nút lá.

λ: tham số điều chỉnh độ lớn của trọng số

1. **Cập nhật mô hình**:
   * Cập nhật giá trị dự đoán bằng cách thêm cây mới:

Trong đó:

* + - η: learning rate.
    - : dự đoán của cây t.

1. **Lặp lại**:
   * Lặp lại cho đến khi đạt số lượng cây tối đa hoặc sai số hội tụ.

**Ưu điểm**

* Hiệu suất cao: Tốc độ huấn luyện nhanh hơn so với các thuật toán boosting truyền thống.
* Chính xác: Kết hợp nhiều cơ chế để giảm lỗi và overfitting.
* Linh hoạt: Hỗ trợ nhiều loại bài toán (hồi quy, phân loại, ranking).
* Tùy chỉnh dễ dàng: Hỗ trợ nhiều tham số để tinh chỉnh (regularization, learning rate, số cây...).

**Nhược điểm**

* Phức tạp: Cần điều chỉnh nhiều tham số để đạt hiệu suất tối ưu.
* Yêu cầu tài nguyên: Tiêu thụ bộ nhớ và CPU cao với tập dữ liệu lớn.
* Dễ bị overfitting nếu không được tinh chỉnh hợp lý.

**LightGBM**

LightGBM (Light Gradient Boosting Machine) là một thuật toán boosting dựa trên cây quyết định, được phát triển bởi Microsoft. Nó được thiết kế để xử lý các bài toán học máy với tốc độ nhanh và hiệu quả cao, đặc biệt phù hợp với tập dữ liệu lớn hoặc có số lượng đặc trưng nhiều.

**Nguyên lý hoạt động**

LightGBM thuộc họ gradient boosting, hoạt động bằng cách xây dựng một chuỗi các mô hình cây quyết định (decision trees). Mỗi cây mới được thêm vào nhằm giảm thiểu sai số của mô hình hiện tại bằng cách tối ưu gradient của hàm mất mát.

Điểm đặc biệt của LightGBM là cách nó xây dựng cây và xử lý dữ liệu:

1. Growth Leaf-wise (phát triển cây theo lá):
   * Thay vì mở rộng cây theo level-wise (mỗi cấp mở rộng đều), LightGBM mở rộng cây theo hướng tối ưu hóa giảm sai số tại các lá tốt nhất (leaf-wise growth).
   * Kết quả là cây có cấu trúc không cân bằng nhưng hiệu quả cao hơn.
2. Histogram-based Algorithm (thuật toán dựa trên histogram):
   * Chuyển đổi giá trị liên tục thành các nhóm (bin) giá trị rời rạc.
   * Giảm đáng kể thời gian tính toán và bộ nhớ cần thiết.
3. Feature Bundling (gom nhóm đặc trưng):
   * Kết hợp các đặc trưng ít liên quan để giảm kích thước dữ liệu.
4. Optimal Split Finding (tìm kiếm chia tách tối ưu):
   * Sử dụng các phương pháp tối ưu để chọn điểm chia cắt tốt nhất, giúp xây dựng cây hiệu quả hơn.

**Thuật toán cụ thể**

Giả sử tập dữ liệu có nnn mẫu (xi,yi)

Khởi tạo mô hình: Bắt đầu với một giá trị dự đoán ban đầu.

Tính gradient và hessian:

Với mỗi mẫu, tính gradient (gi) và hessian (hi) dựa trên hàm mất mát L(y, ӯ):

Xây dựng cây quyết định:

* + Tạo histogram: Chuyển các giá trị liên tục của mỗi đặc trưng thành các bin giá trị rời rạc.
  + Tìm điểm chia tách tốt nhất: Sử dụng gradient và hessian để tìm điểm chia cắt tối ưu trên histogram.
  + Phát triển cây: Chọn lá có mức giảm mất mát (loss reduction) lớn nhất để chia tách tiếp theo (leaf-wise growth).

Cập nhật mô hình:

* + Cập nhật giá trị dự đoán bằng cách thêm cây mới:

Trong đó:

* + - η: learning rate.
    - : dự đoán của cây t.

Lặp lại: Lặp lại cho đến khi đạt số lượng cây tối đa hoặc sai số hội tụ.

**Ưu điểm**

* Tốc độ nhanh:

Thuật toán dựa trên histogram giúp giảm đáng kể thời gian huấn luyện.

Tính toán song song tăng tốc quá trình xử lý.

* Hiệu quả với dữ liệu lớn: Hoạt động tốt trên tập dữ liệu lớn và có nhiều đặc trưng (high-dimensional data).
* Tiết kiệm bộ nhớ: Nhóm các giá trị liên tục thành bin giúp giảm yêu cầu bộ nhớ.
* Tùy chỉnh linh hoạt: Hỗ trợ nhiều tham số điều chỉnh để giảm overfitting và tối ưu hiệu suất.
* Hỗ trợ dữ liệu thiếu: Tự động xử lý dữ liệu bị thiếu mà không cần tiền xử lý.

**Nhược điểm**

* Overfitting: Phát triển cây theo lá (leaf-wise growth) có thể dẫn đến overfitting, đặc biệt với tập dữ liệu nhỏ hoặc nhiễu.
* Cần tinh chỉnh tham số: Để đạt hiệu suất tối ưu, cần điều chỉnh nhiều tham số như.
* Không phù hợp với dữ liệu nhỏ: Với tập dữ liệu nhỏ, các thuật toán khác như Random Forest hoặc XGBoost có thể hoạt động tốt hơn.
* Phức tạp khi triển khai: Dễ bị nhầm lẫn giữa các tham số nếu không quen thuộc.

## 2.2. Các phương pháp đánh giá mô hình

Để đánh giá xem mô hình có dự đoán tốt và chính xác hay không, chúng ta sử dụng 4 độ đo[5] và trực quan đồ thị để so sánh.

* Hệ số xác định (R2): Biểu diễn tỷ lệ phương sai của biến phụ thuộc được giải thích bằng mô hình. R2 là điểm không có thang đo, tức là bất kể giá trị lớn hay nhỏ, giá trị của hệ số xác định sẽ luôn trong khoảng từ 0 đến 1 và càng gần 1 thì mô hình càng tốt.
* Sai số trung bình tuyệt đối (MAE): Là độ lớn trung bình các sai số trong một tập hợp giữa giá trị dự đoán và giá trị thực tế. Độ đo càng nhỏ mô hình càng tốt.
* Sai số bình phương trung bình (MSE): Là một độ đo phổ biến cho các bài toán hồi quy, MSE tìm sai số bình phương trung bình giữa các giá trị được dự đoán và thực tế. Độ đo càng gần 0 mô hình càng tốt.
* Căn bậc hai sai số bình phương trung bình (RMSE): Tương tự MSE nhưng lấy căn bậc hai. RMSE nhạy cảm hơn với các giá trị ngoại lai. Độ đo càng nhỏ càng tốt.

Trong đó: là giá trị thực tế; là giá trị dự đoán, là giá trị trung bình của dữ liệu thực tế, n là số lượng mẫu.

# CHƯƠNG 3: ỨNG DỤNG PHƯƠNG PHÁP VÀ XÂY DỰNG MÔ HÌNH

## 3.1 Công cụ sử dụng và quy trình thu thập dữ liệu

### 3.1.1 Công cụ sử dụng

Môi trường sử dụng:

* Ngôn ngữ lập trình: Python 3.10
* Trình biên dịch: Visual Studio Code
* Nền tảng phát triển: Google Colab
* Thiết bị: Window 11, Ram 8gb, CPU Ryzen 5000 Series, CPU Intel i5 8th

Thư viện sử dụng chính:

* Pandas: Là một thư viện rất phổ biến để làm việc với dữ liệu, chủ yếu là dữ liệu dạng bảng có cấu trúc Dataframe. Pandas Dataframe có thể dễ dàng thay đổi và thao tác. Thư viện có các chức năng hữu ích như xử lý giá trị thiếu, thực hiện các thao tác trên cột và hàng cũng như chuyển đổi dữ liệu.
* Numpy: Là một thư viện mã nguồn mở hỗ trợ các hàm và công cụ giúp tính toán sô học và thống kê.
* Sklearn: Là thư viện học máy mã nguồn mở được xây dựng bằng Python. Nó cung cấp nhiều công cụ và thuật toán cho các nhiệm vụ khác nhau như việc huấn luyện mô hình, đánh giá mô hình, chuẩn hóa dữ liệu.
* Matplotlib – Seaborn: Là thư viện giúp trực quan hóa dữ liệu trong Python. Matplotlib cung cấp các công cụ tạo biểu đồ cơ bản, trong khi đó Seaborn cung cấp nhiều giao diện mặc định và nhiều biểu đồ khác nhau để trực quan hóa thống kê.

### 3.1.2 Quy trình thu thập dữ liệu

**Nguồn dữ liệu**

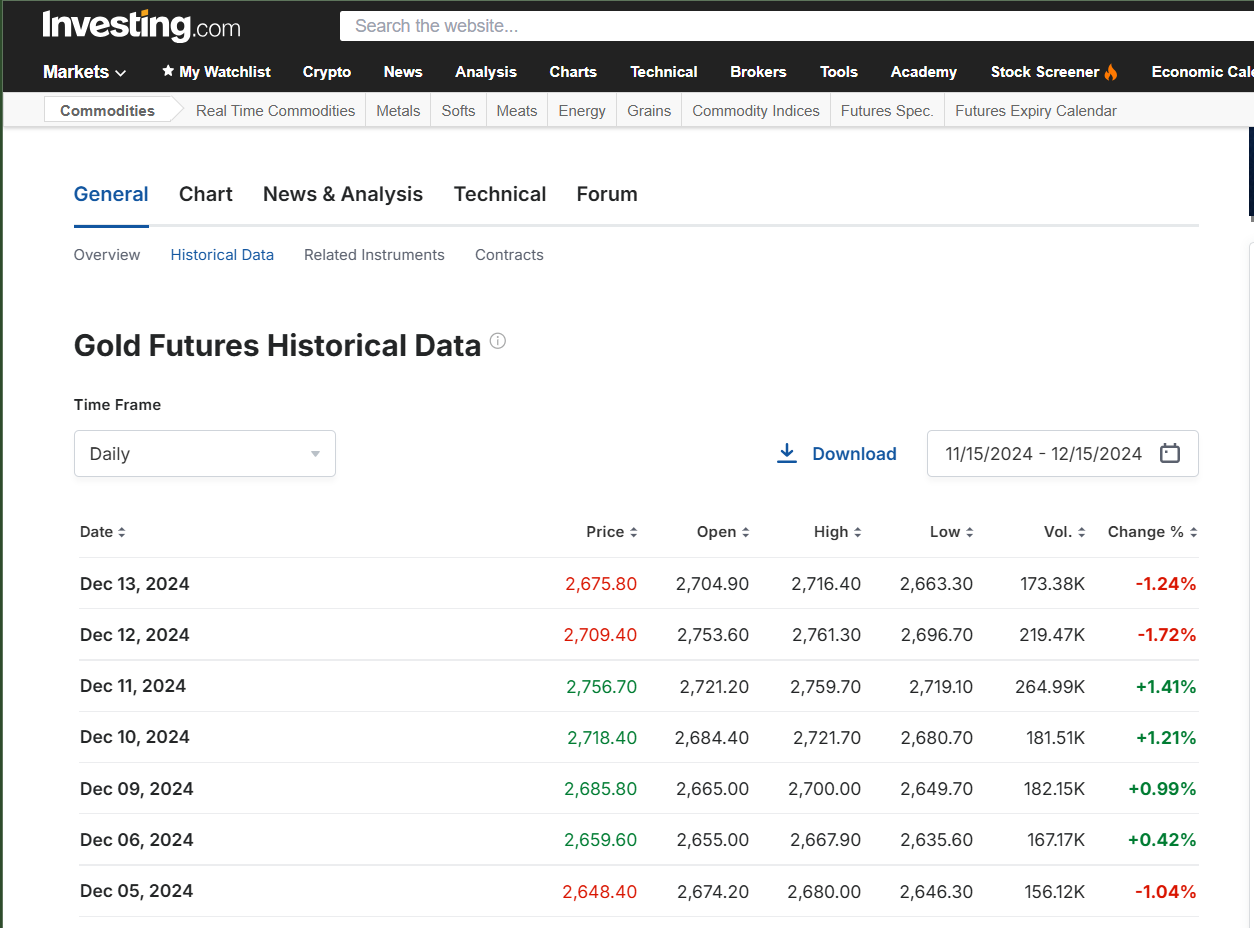
Được thành lập vào năm 2007, Investing.com[6] là một nền tảng cung cấp dữ liệu, bảng báo giá theo thời gian thực, tin tài chính, các bài phân tích trên 250 sàn giao dịch trên khắp thế giới.

Ngoài thông tin về các Thị trường Chứng khoán trên toàn cầu, Investing.com còn cung cấp thông tin về Hàng hóa, Tiền điện tử, các Chỉ số quốc tế, Các loại tiền tệ trên thế giới, Trái phiếu, các Quỹ và Lãi suất, Hợp đồng tương lai và Quyền chọn của các Quỹ giao dịch ngoại hối (ETF).

Nguồn dữ liệu trong đề tài đề tài này, nhóm thực hiện thu thập dữ liệu về giá vàng từ https://www.investing.com/ bằng thư viện BeautifulSoup, cloudscraper và selenium.

**Thực nghiệm thu thập**

Đầu tiên, ta truy cập vào đường dẫn url để dẫn đến trang hiển thị bảng dữ liệu giá vàng như Hình 3.1.



*Hình 3.1: Giao diện chứa dữ liệu giá vàng url dẫn đến*

Bởi vì trang web được bảo về bởi Cloudflare, một hệ thống bảo mật dùng để ngăn chặn các lượt truy cập được nghi là thực hiện bởi bot, ta sẽ vượt qua lớp bảo mật này bằng cloudscraper[7].

url = 'https://www.investing.com/commodities/gold-historical-data'

scraper = cloudscraper.create\_scraper()

options = Options()

driver = Chrome(options=options)

driver.get(url)

Sau khi thành công trong việc vượt qua Cloudflare, ta tìm các phần tử được chỉ định bằng XPath, phân tích cú pháp trang web và tìm bảng dữ liệu bằng class name. Tương ứng với các thông tin cần trích xuất trong Hình 3.2.

element = driver.find\_element(By.XPATH, '//\*[@class="historical-data-v2\_selection-arrow\_\_3mX7U relative flex flex-1 items-center justify-start gap-1 rounded border border-solid border-[#CFD4DA] bg-white bg-select-custom px-3 py-2 text-sm leading-5 text-[#333] shadow-select sm:w-[210px] "]')

element.click()

time.sleep(10);

element = driver.find\_element(By.XPATH, '//\*[@class="flex flex-1 flex-col justify-center text-sm leading-5 text-[#333]"]')

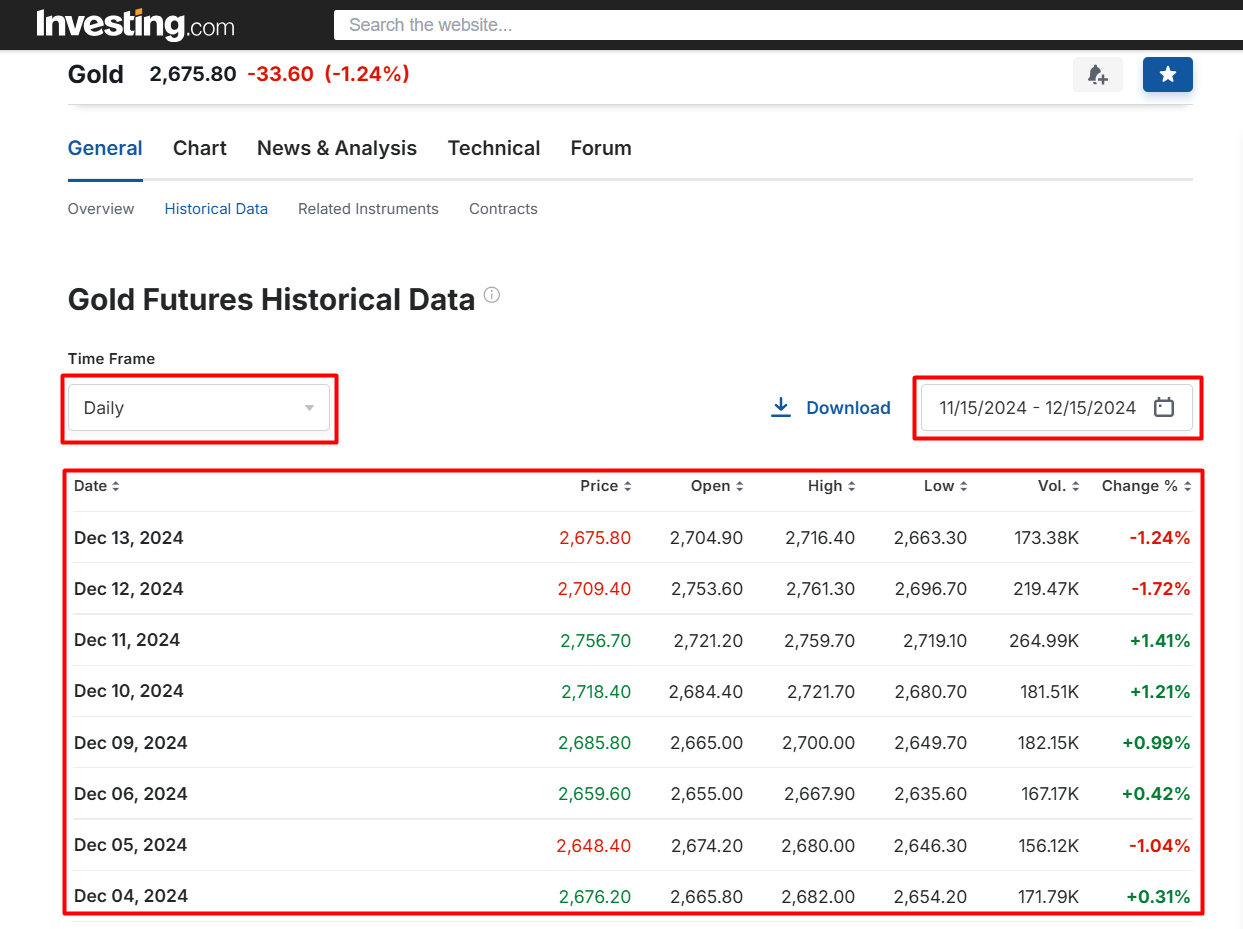
element.click()

time.sleep(60)

soup = BeautifulSoup(driver.page\_source, 'html.parser')

print("\n")

element = soup.find('table', class\_='freeze-column-w-1 w-full overflow-x-auto text-xs leading-4')



*Hình 3.2: Các thông tin cần thu thập*

Cuối cùng, dữ liệu được trích xuất sẽ được ghi vào tệp csv.

with open('Gold\_2003\_2024.csv', 'w', newline='') as csvfile:

    writer = csv.writer(csvfile)

    writer.writerow(['Date', 'Price', 'Open', 'High', 'Low', 'Vol', 'Change%'])

    for row in element.find\_all('tr'):

        cells = row.find\_all('td')

        if cells:

            date = cells[0].text.strip()

            price = cells[1].text.strip()

            open\_price = cells[2].text.strip()

            high = cells[3].text.strip()

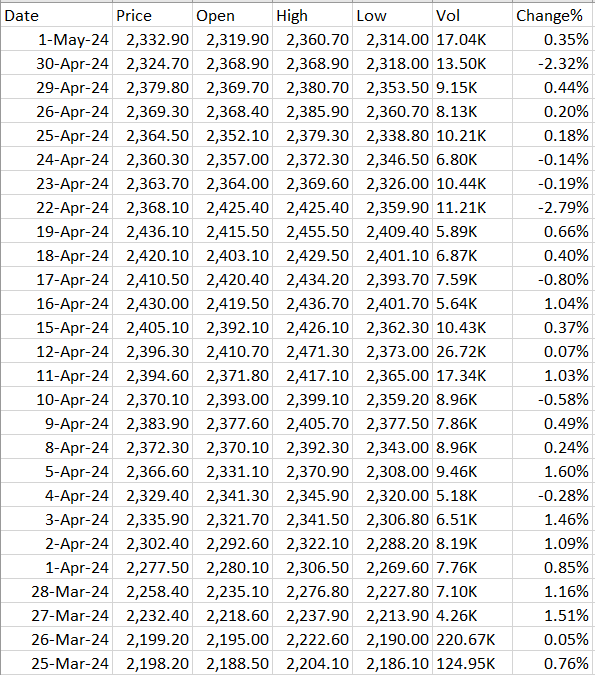
            low = cells[4].text.strip()

            vol = cells[5].text.strip()

            change\_percent = cells[6].text.strip()

            writer.writerow([date, price, open\_price, high, low, vol, change\_percent])

### 3.1.3 Kết quả đạt được



*Hình 3.3: Bộ dữ liệu về giá vàng sau khi thu thập*

Bộ dữ liệu gồm 5416 mẫu với các thuộc tính:

* “Date”: Ngày tháng năm dữ liệu được ghi
* “Price”: Giá đóng phiên giao dịch
* “Open”: Giá mở phiên giao dịch
* “High”: Giá cao nhất phiên giao dịch
* “Low”: Giá thấp nhất phiên giao dịch
* “Vol”: Số lượng giao dịch
* “Change%”: Tỉ lệ chênh lệch giữa giá đóng với giá mở

## 3.3 Tiền xử lý dữ liệu

### 3.3.1 Xử lý giá trị dữ liệu

Sau khi thu thập dữ liệu từ web, tập dữ liệu thô chưa thể đưa ngay vào mô hình học máy được do dữ liệu chưa đồng nhất và đang ở dạng chuỗi chưa phù hợp để đưa vào mô hình. Ta có thể thấy rằng trong cột “Price” các giá trị có chưa dấu “,” điều này khiến ta nhìn qua thì dữ liệu có vẻ là dạng số nhưng bản chất lại là dạng chuỗi. Tương tự với các cột “Open”, “High”, “Low”. Với cột “Vol” thì là ký tự “K”, cột “Change%” là ký tự “%”. Ta sẽ tiến hành loại bỏ các ký tự này.

columns = ['Price', 'Open', 'High', 'Low', 'Vol', 'Change%']

for col in columns:

    data[col]=data[col].str.replace(',','').str.replace('+','').str.replace('%', '').str.replace('K', '')

### 3.3.2 Chuẩn hóa cấu trúc

Với cột “Date” ta thấy rằng thông tin tháng vẫn đang ở dạng chữ, ta sẽ chuyển sang dạng số và chuyển thành dạng datetime.

def convert\_date\_format(date\_str):

    return datetime.strptime(date\_str, '%b %d, %Y').strftime('%m/%d/%Y')

# Áp dụng hàm để chuyển đổi định dạng cột Date

data['Date'] = data['Date'].apply(convert\_date\_format)

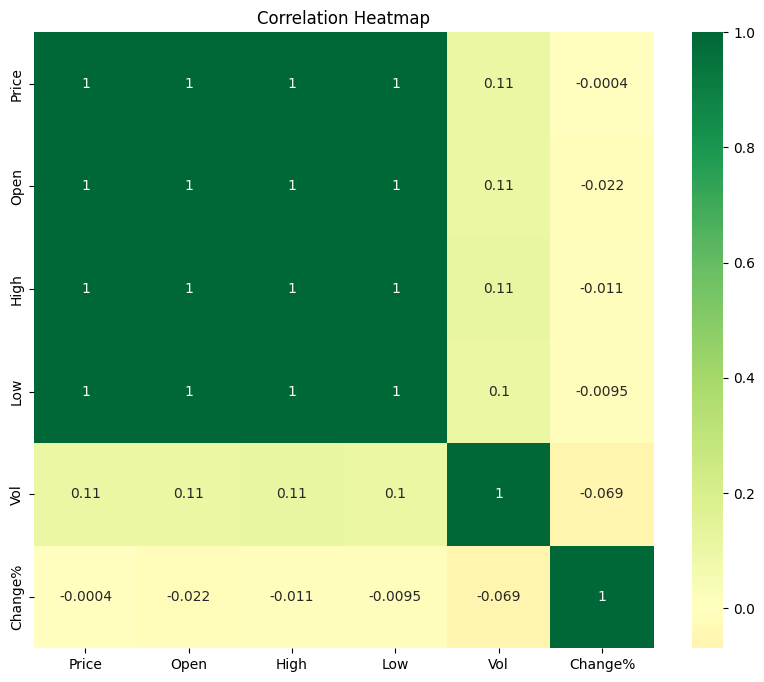
Với các cột còn lại sau khi loại bỏ ký tự thừa, ta sẽ chuyển sang dạng số. Riêng cột “Vol” ký tự “K” thể hiện đơn vị tính hàng nghìn nên ta sẽ nhân các giá trị của cột này với 1000.

data[columns] = data[columns].apply(pd.to\_numeric)

data['Vol'] = data['Vol'] \* 1000

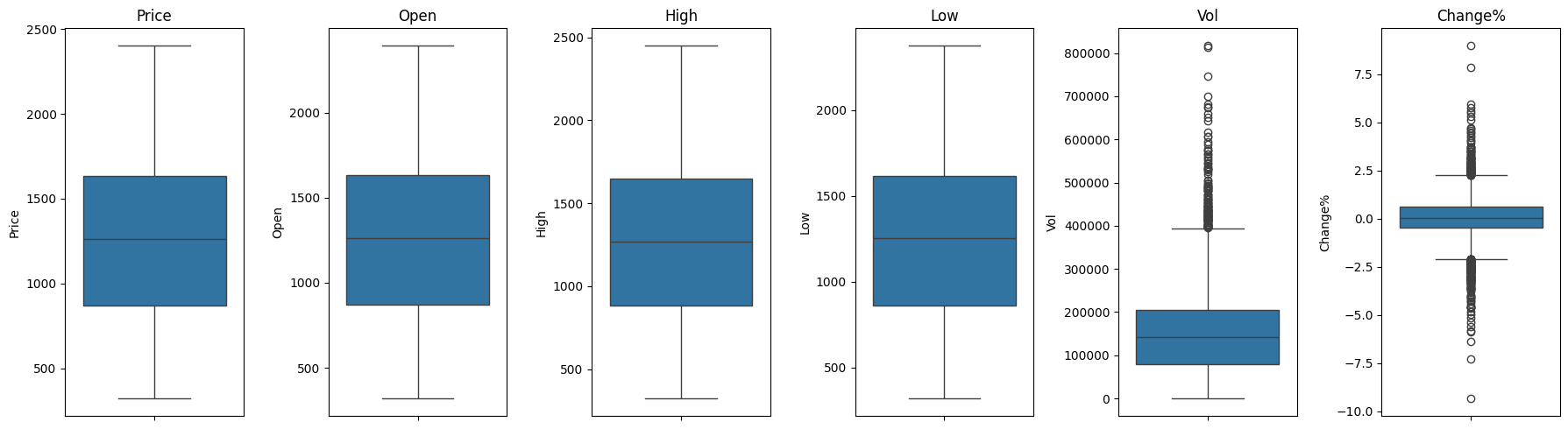
Ngoài ra trong cột “Vol” có chứa duy nhất 1 điểm dữ liệu thiếu nhưng số do số lượng không đáng kể nhóm sẽ điền các điểm dữ liệu thiếu này bằng giá trị trung bình trong cột.

## 3.4 Phân tích dữ liệu



*Hình 3.5: Biểu đồ tương quan heatmap*

Từ biểu đồ trên ta có thấy được rằng các cột “Price”, “Open”, “High”, “Low” có mối tương quan hoàn hảo, tức là các các giá trị sẽ thay đổi, biến động đồng bộ với nhau. Ngược lại ta cũng có thể với chỉ số tương quan như trên biểu đồ, các cột “Vol” và “Change%” sẽ chứa các giá trị ngoại lai và có độ liên quan thấp đến các cột khác trong bộ dữ liệu.



*Hình 3.6: Biểu đồ boxplot cho các cột*

Đúng như dự đoán, các cột “Vol” và “Change%” chứa dữ liệu ngoại lai. Để giữ lại các điểm dữ liệu, nhóm sử dụng Inter-Quartile Range (IQR) để xử lý các giá trị ngoại lai. Một điểm dữ liệu sẽ được coi là ngoại lai nếu thuộc ngưỡng sau:

Trong đó:

x là điểm dữ liệu, Q1 là tứ phân vị thứ nhất, Q3 là tứ phân vị thứ ba, IQR = Q3 - Q1

# Hàm để xử lý ngoại lệ dựa trên IQR cho một cột cụ thể

def handle\_outliers\_iqr(df, column):

    IQR = df[column].quantile(0.75) - df[column].quantile(0.25)

    lower\_limit = df[column].quantile(0.25) - (IQR \* 1.5)

    upper\_limit = df[column].quantile(0.75) + (IQR \* 1.5)

    print(f"Lower limit for {column}: {lower\_limit}")

    print(f"Upper limit for {column}: {upper\_limit}")

    # Xử lý các giá trị ngoại lệ

    df[column] = np.where(df[column] > upper\_limit, upper\_limit,

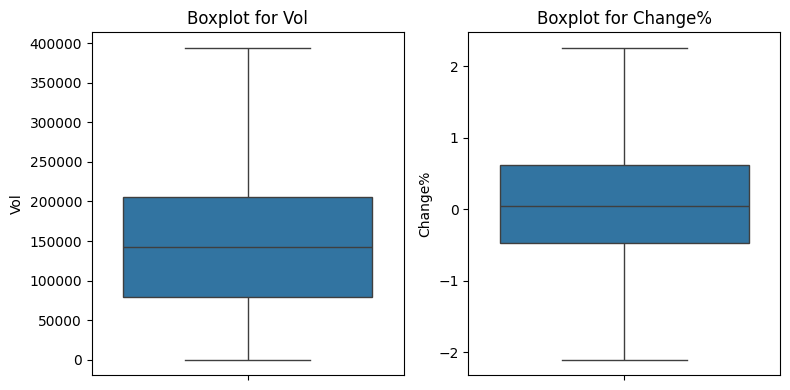
                          np.where(df[column] < lower\_limit, lower\_limit, df[column]))

# Xử lý ngoại lệ cho cột 'Vol'

handle\_outliers\_iqr(data, 'Vol')

# Xử lý ngoại lệ cho cột 'Change%'

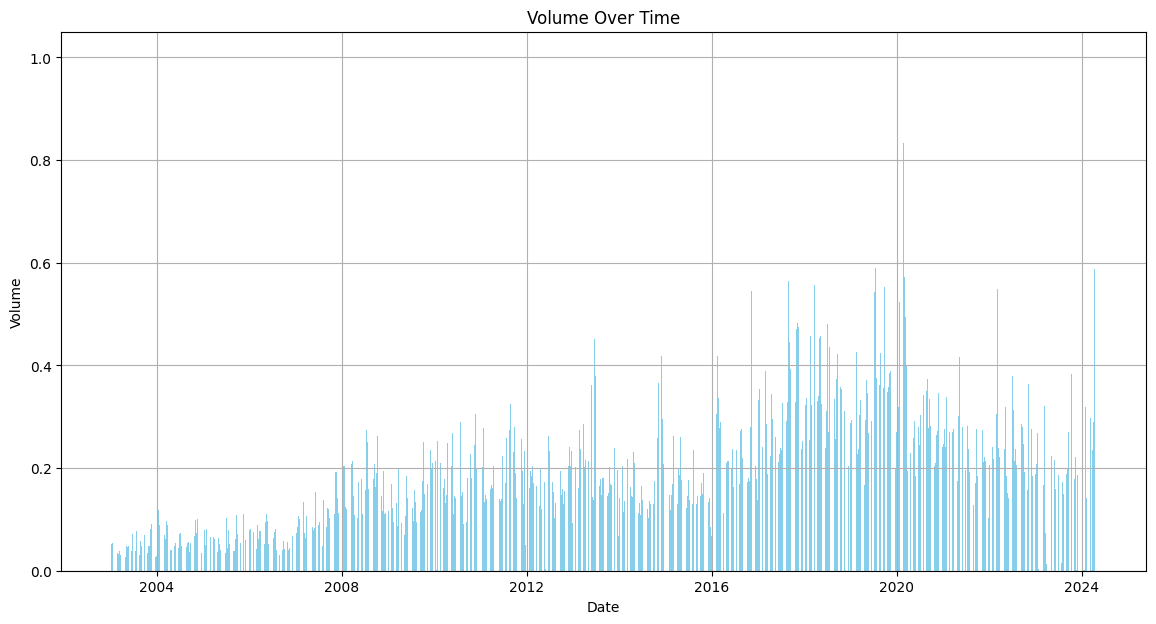
handle\_outliers\_iqr(data, 'Change%')



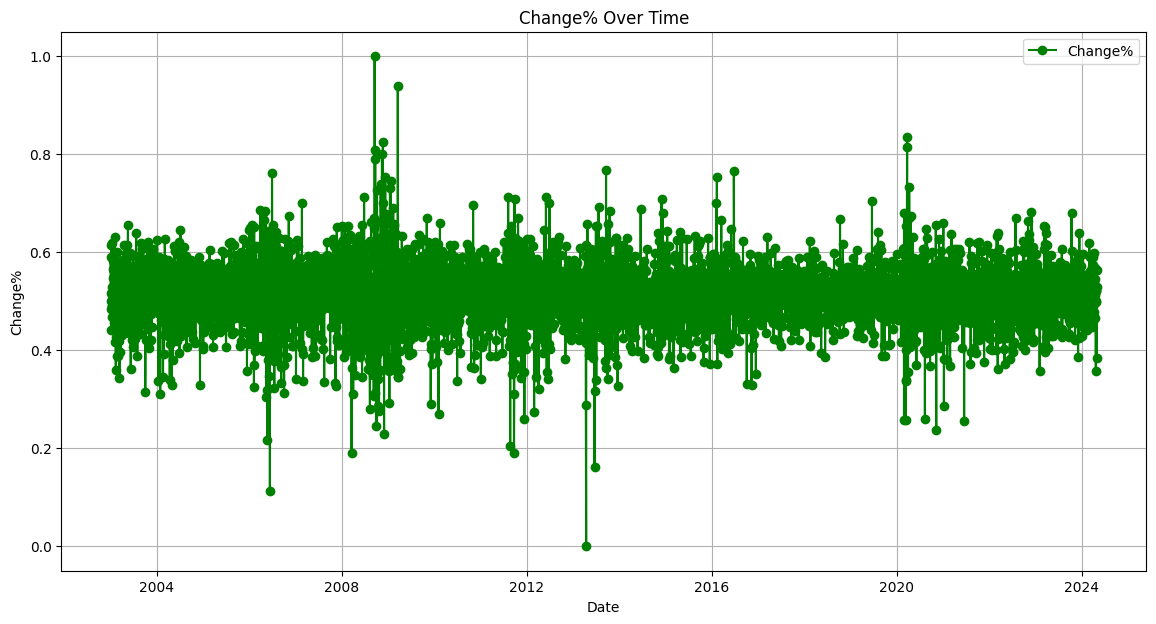
*Hình 3.7: Cột Vol và Change% sau khi xử lý ngoại lai*



*Hình 3.8: Biểu đồ đường cho các cột giá qua thời gian*



*Hình 3.9: Biểu đồ cột Vol qua thời gian*



*Hình 3.10: Biểu đồ tỉ lệ cột Change% qua thời gian*

Từ ba biểu đồ trên ta có thể thấy rằng cả giá vàng ở các hạng mục lẫn khối lượng giao dịch trong khoảng thời gian từ năm 2004-2024 có xu hướng tăng theo thời gian song tỉ lệ chệnh lệch giữa giá đóng và mở không quá biến động.

## 3.5 Xây dựng và huấn luyện mô hình

Từ biểu đồ so sánh sự tương quan dữ liệu hình 3.4, ta thấy rằng 2 cột “Vol” và “Change” có độ tương quan thấp, dùng để huấn luyện có thể gây ảnh hưởng đến mô hình, làm cho mô hình ra kết quả đánh giá không tốt. Nhóm sẽ xây dựng và chạy thử với 2 trường hợp đầy đủ các thuộc tính và loại bỏ “Vol” và”Change”.

Nhóm sẽ tiến hành dự đoán giá đóng. Tập dữ liệu được chia thành 2 tập con gồm tập huấn luyện chiếm 80% và 20% còn lại làm tập kiểm tra.

**3.5.1 Mô hình AdaBoost**

ada\_model = AdaBoostRegressor(estimator=DecisionTreeRegressor(max\_depth=4), n\_estimators=100, random\_state=42)

Mô hình được gọi từ thư viện sklearn với các tham số:

* estimator**:** Mô hình cơ sở (Base Estimator)
* Mô hình cơ sở là thành phần quan trọng trong AdaBoost, chịu trách nhiệm học từ dữ liệu ở từng vòng lặp.
* Trong bài toán này, mô hình cơ sở là DecisionTreeRegressor, một cây quyết định được sử dụng để học các mối quan hệ phi tuyến tính trong dữ liệu. Độ sâu tối đa nhóm đặt là 4.
* Vai trò:
  + Đảm bảo khả năng xử lý tốt các mẫu dữ liệu đơn giản ở mỗi vòng lặp.
  + Nếu cây quyết định quá phức tạp, AdaBoost có thể bị overfitting.
* n\_estimators: Số lượng mô hình con (Number of Weak Learners)
* Tham số này xác định số vòng lặp mà AdaBoost sẽ thực hiện để huấn luyện các mô hình con. Trong bài toán này AdaBoost sẽ xây dựng 100 cây quyết định liên tiếp.
* Vai trò: Số lượng cây lớn hơn thường tăng hiệu quả dự đoán, nhưng giá trị quá cao sẽ làm tăng chi phí tính toán và nguy cơ overfitting.
* random\_state: Trạng thái ngẫu nhiên (Random State)
* Tham số này đảm bảo kết quả huấn luyện và kiểm tra được lặp lại giống nhau mỗi khi chạy lại mã.
* Vai trò: Giúp kiểm soát sự ngẫu nhiên trong quá trình chia dữ liệu hoặc trong mô hình hóa, thuận tiện cho việc so sánh kết quả giữa các lần chạy.

**3.5.2 Mô hình XGBoost**

xgb\_model = xgb.XGBRegressor(objective='reg:squarederror', n\_estimators=100, random\_state=42)

Mô hình được gọi từ thư viện sklearn với các tham số:

* objective: Hàm mục tiêu. Xác định loại bài toán và hàm mất mát mà mô hình sẽ tối ưu hóa trong quá trình huấn luyện.

Trong bài toán nhóm sử dụng: objective='reg:squarederror' (bình phương sai số).

* n\_estimators: Số lượng cây quyết định
* Ý nghĩa: Quy định số lượng cây quyết định (learners) trong quá trình xây dựng mô hình. Ở đây nhóm sử dụng n\_estimators=100.
* Vai trò: Số lượng lớn hơn thường giúp cải thiện độ chính xác, nhưng có thể dẫn đến việc mô hình bị overfitting nếu không được kiểm soát.
* Random\_state: đảm bảo kết quả huấn luyện và kiểm tra được lặp lại giống nhau mỗi khi chạy lại mã.

**3.5.3 Mô hình LightGBM**

lgb\_model = lgb.LGBMRegressor(

    boosting\_type='gbdt',

    max\_depth=6,

    num\_leaves=31,

    n\_estimators=100,

    learning\_rate=0.1,

    random\_state=42

)

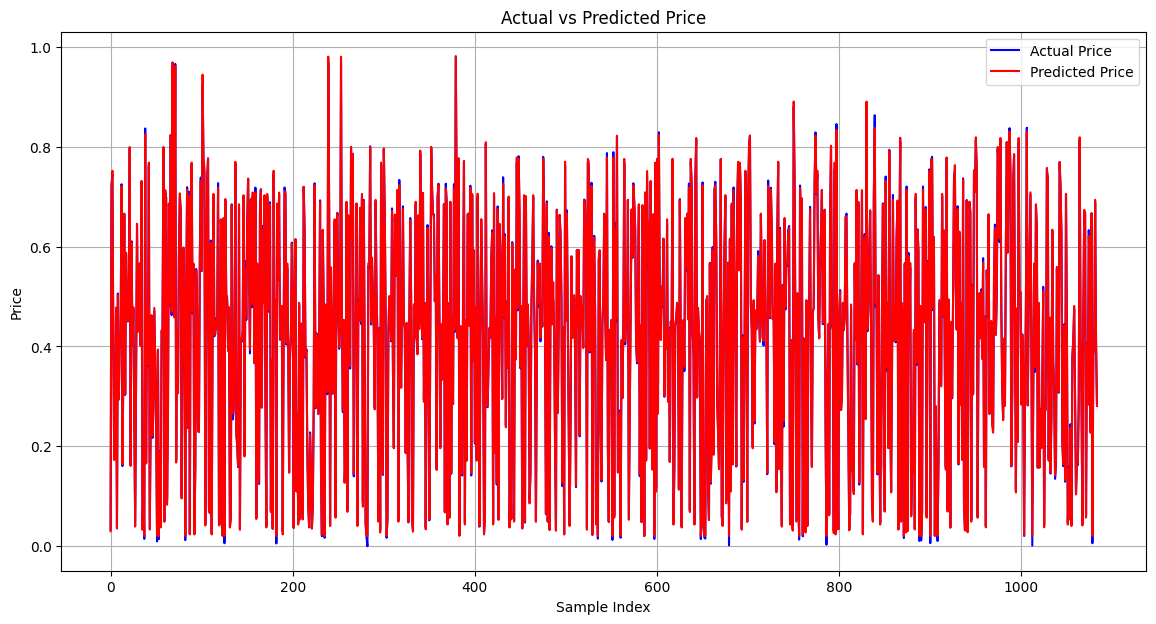
Mô hình được gọi từ thư viện sklearn với các tham số:

* boosting\_type: Loại thuật toán boosting
* Ý nghĩa:
  + Quy định phương pháp boosting mà mô hình sử dụng.
  + Trong bài toán: boosting\_type='gbdt': Phương pháp boosting truyền thống, thường được sử dụng cho các bài toán tổng quát
* max\_depth: Độ sâu tối đa của cây
* Ý nghĩa:
  + Quy định độ sâu tối đa của mỗi cây quyết định.
  + Trong bài toán: max\_depth = 6.
* Vai trò: Độ sâu lớn hơn cho phép mô hình học được các mẫu phức tạp hơn, nhưng dễ dẫn đến overfitting.
* num\_leaves: Số lượng lá (leaves)
* Ý nghĩa:
  + Quy định số lượng nút lá tối đa mà mỗi cây quyết định có thể tạo ra.
  + Trong bài toán: num\_leaves=31.
* Vai trò:
  + Số lượng lá lớn hơn cho phép mô hình phân tách dữ liệu chi tiết hơn, nhưng cũng tăng nguy cơ overfitting.
  + Quy tắc: , đảm bảo tối ưu hóa hiệu suất.
* n\_estimators: Số lượng cây quyết định
* Ý nghĩa:
  + Xác định số lượng cây quyết định sẽ được xây dựng trong mô hình boosting.
  + Trong bài toán: n\_estimators=100.
* Vai trò:
  + Tăng số lượng cây giúp cải thiện độ chính xác, nhưng cũng làm tăng thời gian huấn luyện.
  + Thường được kết hợp với learning\_rate để tìm điểm cân bằng giữa tốc độ học và độ chính xác.
* learning\_rate: Tốc độ học
* Ý nghĩa:
  + Quy định mức độ điều chỉnh trọng số của từng cây trong mô hình.
  + Trong bài toán: learning\_rate=0.1.
* Vai trò:
  + Giá trị nhỏ hơn (ví dụ: 0.01 - 0.1) giúp quá trình học ổn định hơn và giảm overfitting.
  + Giá trị lớn hơn có thể làm cho mô hình hội tụ nhanh nhưng dễ bỏ qua các mẫu phức tạp.
* random\_state: Trạng thái ngẫu nhiên. Đảm bảo tính tái lập của quá trình huấn luyện và dự đoán bằng cách kiểm soát các yếu tố ngẫu nhiên.

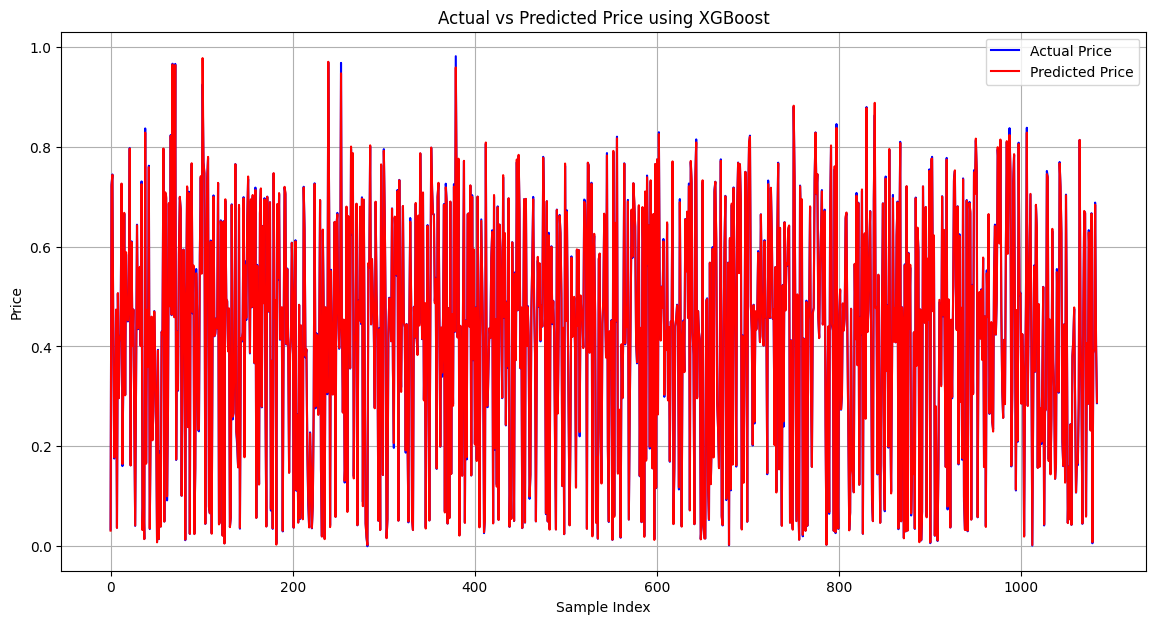
# CHƯƠNG 4: KẾT QUẢ VÀ ĐÁNH GIÁ MÔ HÌNH

## 4.1 Kết quả

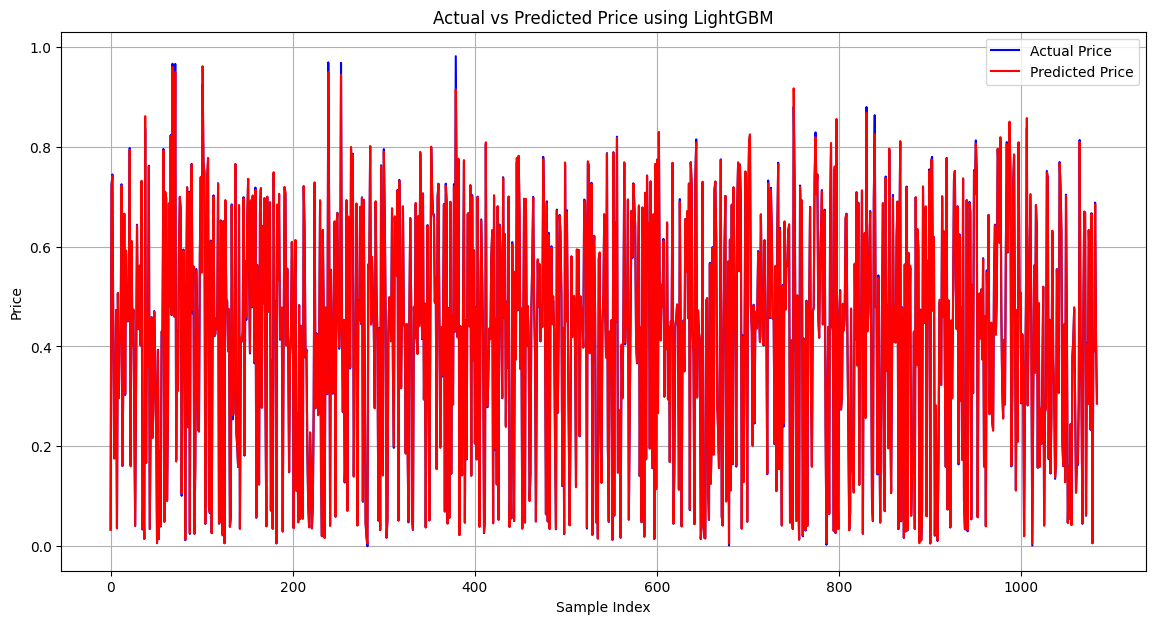
Trong phần này, ta sẽ trực quan kết quả thực tế với kết quả dự đoán của các mô hình dã sử dụng ở phần trước.



*Hình 4.1: Biểu đồ trực quan giá trị thực tế với dự đoán mô hình AdaBoost*



*Hình 4.2: Biểu đồ trực quan giá trị thực tế với dự đoán mô hình XGBoost*



*Hình 4.3: Biểu đồ trực quan giá trị thực tế với dự đoán mô hình LightGBM*

Nhìn qua các biều đồ thì ta có thể thấy tất cả các mô hình được sử dụng đều dự đoán cho ra kết quả đúng với xu hướng biến động của giá trị thực. Để rõ hơn ta hãy sử dụng các độ đo đánh giá mô hình.

## 4.2 Đánh giá mô hình

Trong phần này, ta sẽ trình bày kết quả các thử nghiệm với các độ đo của tất cả mô hình. Sử dụng các độ đo R2, MAE, MSE, RMSE để đánh giá độ chính xác của mô hình.

Huấn luyện khi đầy đủ các thuộc tính:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | R2 | MAE | MSE | RMSE |
| AdaBoost | 0.9995500074 | 0.0038165563 | 0.0000258424 | 0.0050835442 |
| XGBoost | 0.9997946080 | 0.0021295822 | 0.0000117954 | 0.0034344378 |
| LightGBM | 0.9997334262 | 0.0018725070 | 0.0000153089 | 0.0039126651 |

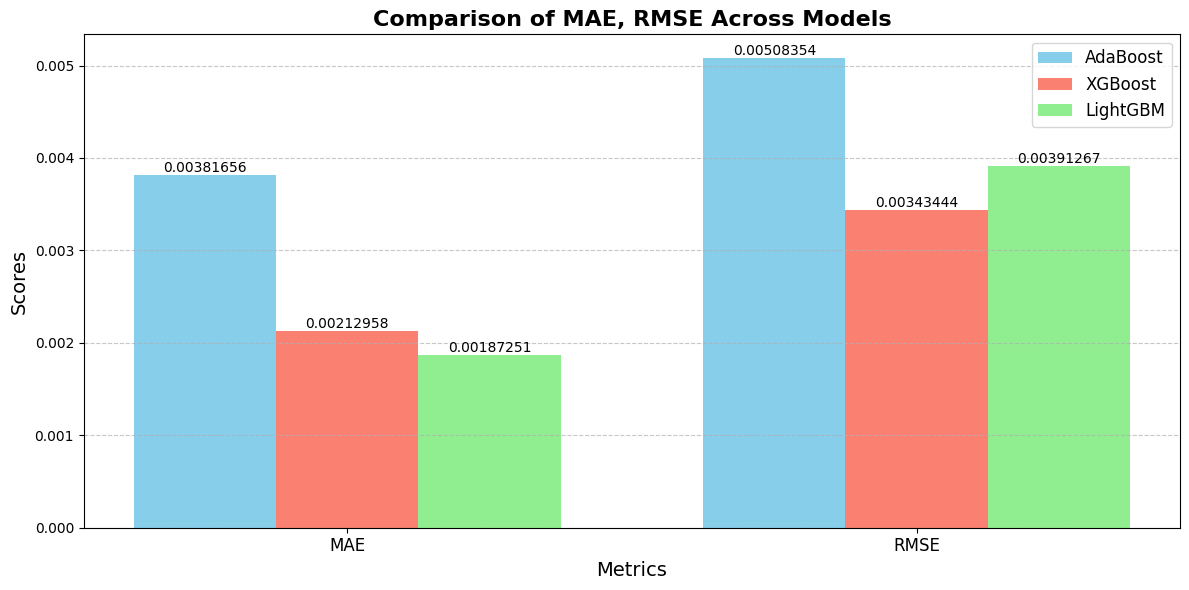
Huấn luyện khi bỏ “Vol”, “Change”:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | R2 | MAE | MSE | RMSE |
| AdaBoost | 0.9994318151 | 0.0044587943 | 0.0000326300 | 0.0057122706 |
| XGBoost | 0.9999207299 | 0.0010910685 | 0.0000045524 | 0.0021336272 |
| LightGBM | 0.9998788546 | 0.0010378403 | 0.0000069572 | 0.0026376515 |

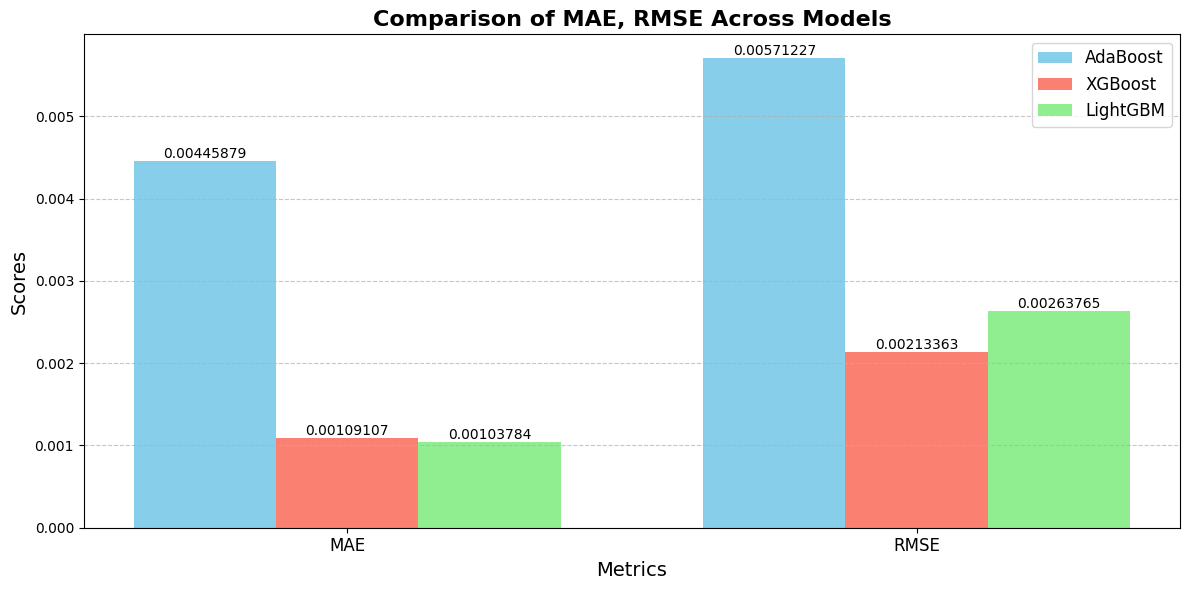
Từ kết quả đánh giá mô hình bằng các độ đo, nhóm thấy rằng mặc dù chênh lệch giữa các chỉ số giữa 2 trường hợp không quá lớn nhưng 2 mô hình XgBoost và LightGBM sẽ cho ra kết quả có phần tốt hơn khi loại bỏ hai thuộc tính “Vol” và “Change” khi mà cả bốn độ đo đều cho ra chỉ số tốt hơn. Mô hình AdaBoost chỉ có 2 chỉ số R2 và MSE tốt hơn khi giữ lại hai cột “Vol” và “Change”.

Để dễ dàng quan sát hơn nhóm trực quan các độ đo trên bảng thành biểu đồ.

|  |  |
| --- | --- |
| *Hình 4.4.1: R2 khi đầy đủ thuộc tính* | *Hình 4.4.2: R2 khi loại bỏ thuộc tính* |



*Hình 4.5.1: MAE, RMSE khi đầy đủ thuộc tính*

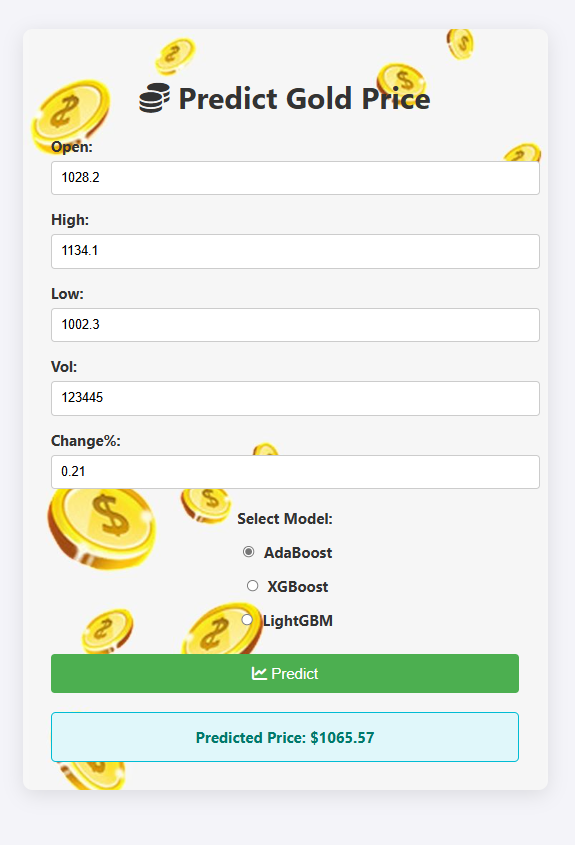


*Hình 4.5.2: MAE, RMSE khi loại bỏ thuộc tính*

|  |  |
| --- | --- |
| *Hình 4.6.1: MSE khi đầy đủ thuộc tính* | *Hình 4.6.2: MSE khi loại bỏ thuộc tính* |

## 4.3 Xây dựng giao diện ứng dụng

Để xây dựng giao diện demo, nhóm sử dụng Flask. Đây là một dạng Micro Framework cơ bản được viết bằng ngôn ngữ lập trình Python, được tạo ra bởi Armin Ronacher, một nhà phát triển phần mềm người Đức vào năm 2010. Công nghệ này thường được sử dụng để xây dựng trang web từ những ứng dụng đơn giản đến những hệ thống phức tạp hơn.



*Hình 4.7: Giao diện web dự đoán giá vàng*

# KẾT LUẬN

Đầu tư vào vàng đã phát triển theo thời gian dưới các hình thức như mua trang sức, hoặc bang các mua tiền vàng và vàng miếng. Trong lịch sử, vàng luôn giữ được giá trị của nó[5] và được sử dụng làm thước đo sức mạnh tiền tệ của một quốc gia. Sau khi tìm hiểu và hoàn thiện báo cáo này, tôi đã có thể áp dụng các mô hình học máy AdaBoost, XGBoost, LightGBM cho bài toán dự đoán giá vàng. Với việc thử các mô hình với các trường hợp đã nêu, tôi thấy rằng mô hình XGBoost và LightGBM cho ra kết quả dự đoán đều khá tốt. tôi cũng đã xây dựng giao diện để dự đoán giá vàng.

# TÀI LIỆU THAM KHẢO

[1]E. A. Enjoy Learning, "*Supervised, Unsupervised and Semi-supervised Learning*

*with Real-life Usecase*" 2023. [Online]. Available:

https://www.enjoyalgorithms.com/blogs/supervised-unsupervised-andsemisupervised-learning

[2] Geeksforgeeks, "*Regression Metrics*," 2023. [Online]. Available:

https://www.geeksforgeeks.org/regression-metrics/.

[3] investing.com. [Online]. Available: <https://www.investing.com/commodities/gold-historical-data>

[4] ZenRows, "*How to Bypass Cloudflare in Python*," 2023. [Online]. Available:

https://www.zenrows.com/blog/bypass-cloudflare-python#what-is-cloudflarebot-manager

[5] J. Jagerson and S. W. Hansen, “All about investing in gold”, McGraw-Hill Publishing, 2011