TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA CÔNG NGHỆ THÔNG TIN

TIỂU LUẬN CUỐI KỲ

HỌC PHẦN: KHOA HỌC DỮ LIỆU

ĐỀ TÀI: DỰ ĐOÁN GIÁ ĐỒNG HỒ

Họ và tên	Mã sinh viên
Nguyễn Quốc Thành	102200154
Huỳnh Lê Đắc Tài	102200152
Bùi Văn Huy	102200133
Nguyễn Hưng	102200132

ĐÀ NẴNG, 06/2022

TÓM TẮT

Bộ dữ liệu bao gồm các thông tin, thông số kỹ thuật, giá bán của đồng hồ, được thu thập từ 3 trang web bán sản phẩm ở Việt Nam. Dữ liệu này được dùng để dự đoán giá đồng hồ với các thông số kỹ thuật được đưa ra. Dữ liệu sau khi thu thập là các chuỗi ký tự trong đó có các đại lượng cần dùng cho mô hình dự đoán. Khi làm sạch dữ liệu, cần trả về đúng kiểu dữ liệu, thay thế các giá trị dữ liệu trống. Để lựa chọn đặc trưng phù hợp cho mô hình dự đoán, sau khi xử lý dữ liệu ngoại lệ cũng như chuẩn hóa dữ liệu, ta tiến hành trực quan hóa sự tương quan của các đặc trưng so với đặc trưng mục tiêu. Và chọn lựa ra các đặc trưng phù hợp cho mô hình dự đoán. Mô hình huấn luyện được xây dựng dựa trên các mô hình như, Random Forest Regressor và mô hình XGBoost Regression. Đánh giá hiệu quả của mô hình dự đoán sử dụng RMSE, MAE và R2.

BẢNG PHÂN CÔNG NHIỆM VỤ

Sinh viên thực hiện	Các nhiệm vụ	Tiến độ
Nguyễn Quốc Thành	- Thu thập dữ liệu - Làm sạch dữ liệu, xử lý dữ liệu trống	- Đã hoàn thành - Đã hoàn thành
Huỳnh Lê Đắc Tài	- Lựa chọn đặc trưng - Mô hình hóa dữ liệu	- Đã hoàn thành - Đã hoàn thành
Bùi Văn Huy	- Đánh giá hiệu quả mô hình - Mô hình hóa dữ liệu	- Đã hoàn thành - Đã hoàn thành
Nguyễn Hưng	- Thống kê mô tả trực quan về dữ liệu - Chuẩn hóa dữ liệu	- Đã hoàn thành - Đã hoàn thành

MỤC LỤC

TÓM TẮT **1**
BẢNG PHÂN CÔNG NHIỆM VỤ **2**
1. Giới thiệu **6**
2. Thu thập và mô tả dữ liệu **6**
2.1. Thu thập dữ liệu 6
2.1.1 Nguồn dữ liệu 6
2.1.2 Công cụ thu thập 6
2.1.3 Cách thức sử dụng công cụ 7
2.2. Mô tả dữ liệu 11
3. Trích xuất đặc trưng **13**
3.1. Loại bỏ các hàng và các cột dữ liệu không cần thiết 13
3.2. Làm sạch dữ liệu với đúng kiểu dữ liệu 13
3.3. Xử lý dữ liệu trống 14
3.4. Chuyển các dữ liệu phân loại thành dữ liệu dạng số 15
3.5. Chuẩn hóa dữ liệu 15
3.6. Độ tương quan giữa các đặc trưng với mục tiêu 16
4. Mô hình hóa dữ liệu **17**
4.1. Random Forest Regression 17
4.2. Mô hình Hồi quy của thư viện XGBoost - XGBoost Regression 19
4.3. So sánh hiệu quả của các mô hình 21
5. Dự đoán và kết luận **23**
5.1. Dự đoán 23
5.2. Kết luận 24
5.2.1. Thống kê mô tả trực quan về dữ liệu 24
5.2.2. Làm sạch dữ liệu 24
5.2.3. Mô hình hóa dữ liệu 24

DANH SÁCH HÌNH ẢNH

STT	Tên hình	Trang
01	Mô hình Web sử dụng Beautiful Soup	07
02	Kiểm tra các tài nguyên trong website của donghohaitrieu.com	07
03	Code lấy thẻ <a> và lưu từ trang web	08
04	Các link lấy được	08
05	Cấu trúc của trang chi tiết sản phẩm	09
06	Code lấy thông tin chi tiết sản phẩm	10
07	Lưu dữ liệu vào file csv	10
08	Số lượng dữ liệu trống của toàn bộ dữ liệu thô	11
09	Phân bố dữ liệu của trường dự đoán Price	12
10	Biểu đồ hộp của cột Price	12
11	Dữ liệu sau khi trả về đúng kiểu dữ liệu	14
12	Dữ liệu sau khi xử lý dữ liệu trống	14
13	Dữ liệu sau khi chuyển dữ liệu phân loại thành dạng số	15
14	Dữ liệu huấn luyện sau khi chuẩn hóa dữ liệu	16
15	Sự tương quan giữa các đặc trưng dạng số với đặc trưng mục tiêu	16
16	Sự tương quan giữa tất cả đặc trưng với đặc trưng mục tiêu	17
17	Sự chênh lệch giá cả dự đoán so với thực tế của Mô hình Random Forest Regression	18
18	Sự chênh lệch giá cả dự đoán so với thực tế của Mô hình XGBoost Regression	21

DANH SÁCH BẢNG BIỂU

STT	Tên bảng	Trang
01	Bảng giá trị các metrics của mô hình Random Forest Regression	19
02	Bảng tham số của mô hình	20
03	Bảng giá trị các metrics của mô hình XGBoost Regression	21
04	Bảng số liệu của các metrics giữa 2 mô hình	21

NỘI DUNG BÁO CÁO

1. Giới thiệu

Đề tài "Dự đoán giá đồng hồ" là một lĩnh vực nghiên cứu trong lĩnh vực học máy và dự đoán. Trong thời đại công nghệ ngày nay, việc dự đoán giá trị của các loại đồng hồ dựa trên các thông tin và đặc điểm của chúng đang thu hút sự quan tâm của nhiều nhà nghiên cứu và người tiêu dùng. Mục tiêu chính của nghiên cứu này là phát triển một mô hình dự đoán giá đồng hồ chính xác. Bằng cách sử dụng các thuật toán và phương pháp học máy tiên tiến, mô hình sẽ dự đoán giá trị của các loại đồng hồ dựa trên các yếu tố quan trọng như thương hiệu, loại máy, vật liệu và chất liệu, tính năng đặc biệt và tình trạng của đồng hồ.Ngoài ra, mô hình này cũng giúp cho khách hàng xác định được giá tốt nhất cho đồng hồ họ muốn mua với các thông số cụ thể.

2. Thu thập và mô tả dữ liệu

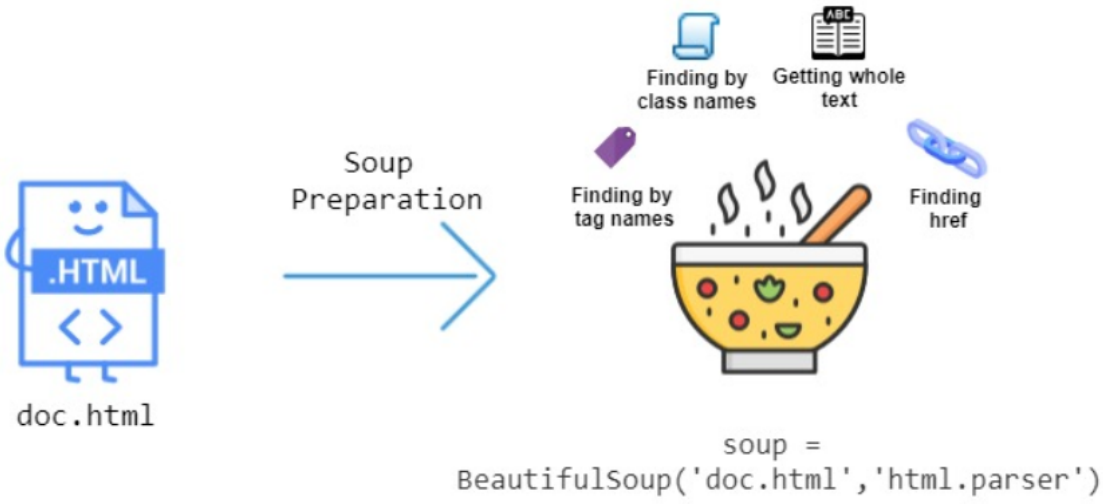
2.1. Thu thập dữ liệu

2.1.1 Nguồn dữ liệu

- Dữ liệu được thu thập từ 3 trang web bán đồng hồ ở Việt Nam.
    - danawatch.vn
    - donghohaitrieu.com
    - dangquangwatch.vn

2.1.2 Công cụ thu thập

- Công cụ đã sử dụng để thu nhập dữ liệu là thư viện **Beautiful Soup**.
- **Beautiful Soup** là một thư viện Python phổ biến và mạnh mẽ được sử dụng để phân tích và trích xuất dữ liệu từ các trang web. Nó cung cấp một cách dễ dàng để phân tích cú pháp HTML và XML, giúp tìm kiếm, truy cập và trích xuất thông tin từ các thành phần của trang web như thẻ, lớp, id…
- Thư viện **Beautiful Soup** giúp tách riêng logic phân tích dữ liệu từ việc tải và xử lý cú pháp HTML/XML, giúp đơn giản hóa quá trình trích xuất thông tin từ trang web và tăng cường hiệu suất và linh hoạt trong việc thu thập dữ liệu.

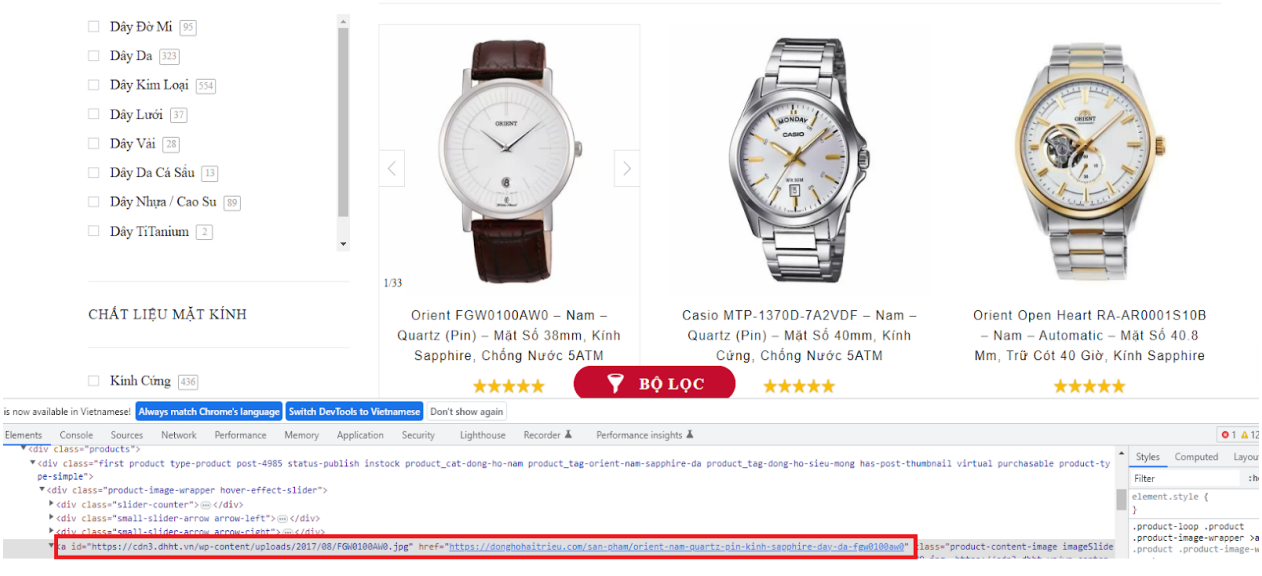


Hình 01: Mô hình Web sử dụng Beautiful Soup

2.1.3 Cách thức sử dụng công cụ

Ví dụ minh họa cho trang donghohaitrieu.com

Bước 1: Truy cập vào trang web và tìm các thẻ <a> có chứa các địa chỉ trỏ tới từng trang chi tiết của sản phẩm.



Hình 02: Kiểm tra các tài nguyên trong website của donghohaitrieu.com

Bước 2 : Lấy các thẻ <a> vừa tìm được và lưu vào mảng

donghohaitrieu.com

```
import bs4
import pandas as pd
# thư viện để call api
import requests
headers = {
    'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/58.0.3029.110 Sa

apiURL ="https://donghohaitrieu.com/danh-muc/dong-ho-nam/page/{}"

def getSoup(url):
        req = requests.get(url,
                           headers=headers)  # URL of the website which you want to scrape
        content = req.content  # Get the content
        soup = bs4.BeautifulSoup(content, 'html.parser')
        # print('soup', soup)
        desc = soup.find('div', {'class':'product-loop products-grid product-count-3'})

        links = desc.find_all('a', {'class':'product-content-image imageSlider'})
        # print(links)
        # dem = 0
        for link in links :
          allLinks.append(link.get('href'))
```

```
# Chứa tất cả link sản phẩm
allLinks=[]
# lấy ra tất cả các link từ trang 0 đến trang 57
for i in range(58):
  getSoup(apiURL.format(i))
```

```
print("Number of link: ",len(allLinks))
```

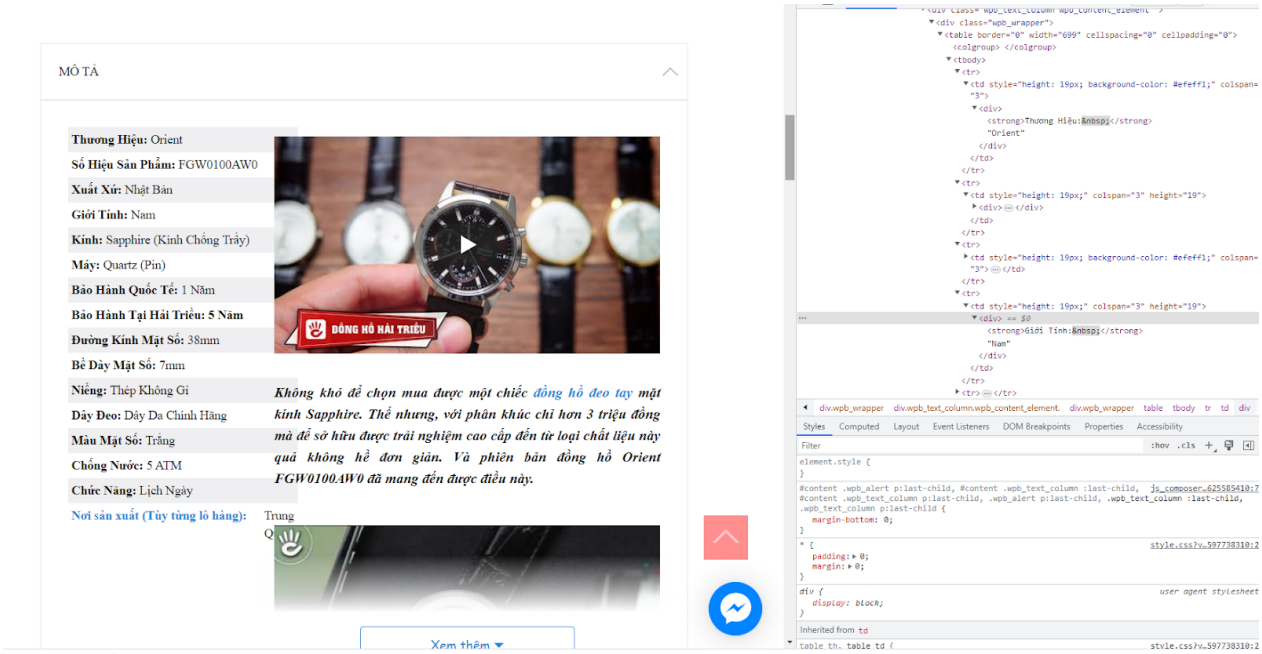
```
Number of link:  1464
```

Hình 03: Code lấy thẻ <a> và lưu từ trang web

```
1186 https://donghohaitrieu.com/san-pham/casio-aq-s810w-1a2vdf-nam-tough-solar-nang-luong-anh-sang-day-cao-su
1187 https://donghohaitrieu.com/san-pham/casio-aq-s810w-2avdf-nam-tough-solar-nang-luong-anh-sang-day-cao-su
1188 https://donghohaitrieu.com/san-pham/casio-ae-1200wh-1bvdf-nam-kinh-nhua-quartz-pin-day-cao-su
1189 https://donghohaitrieu.com/san-pham/citizen-nh8373-88a-nam-automatic-tu-dong-day-kim-loai
1190 https://donghohaitrieu.com/san-pham/g-shock-gst-s300g-1a1dr-nam-tough-solar-nang-luong-anh-sang-day-cao-su
1191 https://donghohaitrieu.com/san-pham/g-shock-gax-100msa-2adr-nam-quartz-pin-day-cao-su
1192 https://donghohaitrieu.com/san-pham/longines-l2-628-4-77-6-nam-kinh-sapphire-automatic-tu-dong-day-kim-loai
1193 https://donghohaitrieu.com/san-pham/casio-efb-510jbl-7avdr-nam-kinh-sapphire-quartz-pin-day-da
1194 https://donghohaitrieu.com/san-pham/op-58012dms-t-dp-04-nam-kinh-sapphire-quartz-pin-day-kim-loai
1195 https://donghohaitrieu.com/san-pham/tissot-t109-410-33-031-00-nam-kinh-sapphire-quartz-pin-day-kim-loai
1196 https://donghohaitrieu.com/san-pham/tissot-t109-407-11-031-00-nam-kinh-sapphire-automatic-tu-dong-day-kim-loai
1197 https://donghohaitrieu.com/san-pham/skagen-skw6352-nam-quartz-pin-day-da
1198 https://donghohaitrieu.com/san-pham/seiko-srpb67j1-nam-kinh-sapphire-automatic-tu-dong-day-da
1199 https://donghohaitrieu.com/san-pham/seiko-srpb65j1-nam-kinh-sapphire-automatic-tu-dong-day-da
1200 https://donghohaitrieu.com/san-pham/seiko-srpb63j1-nam-kinh-sapphire-automatic-tu-dong-day-da
1201 https://donghohaitrieu.com/san-pham/seiko-srp775k1-nam-automatic-tu-dong-day-kim-loai
1202 https://donghohaitrieu.com/san-pham/orient-faa02001b9-nam-automatic-tu-dong-day-kim-loai
1203 https://donghohaitrieu.com/san-pham/longines-l4-921-4-72-6-nam-kinh-sapphire-automatic-tu-dong-day-kim-loai
1204 https://donghohaitrieu.com/san-pham/longines-l1-611-4-52-2-nam-kinh-sapphire-automatic-tu-dong-day-da
1205 https://donghohaitrieu.com/san-pham/frederique-constant-fc-225st5b5-nam-kinh-sapphire-quartz-pin-day-da
1206 https://donghohaitrieu.com/san-pham/fossil-fs5305-nam-quartz-pin-day-da
1207 https://donghohaitrieu.com/san-pham/claude-bernard-53007-37rm-air-nam-kinh-sapphire-quartz-pin-day-kim-loai
1208 https://donghohaitrieu.com/san-pham/citizen-np1014-51e-nam-kinh-sapphire-automatic-tu-dong-day-kim-loai
1209 https://donghohaitrieu.com/san-pham/citizen-bm7375-18h-nam-kinh-sapphire-eco-drive-nang-luong-anh-sang-day-da
1210 https://donghohaitrieu.com/san-pham/citizen-bm7370-11a-nam-kinh-sapphire-eco-drive-nang-luong-anh-sang-day-da
```

Hình 04: Các link lấy được

Bước 3: Phân tích cấu trúc trang web của từng sản phẩm



Hình 05: Cấu trúc của trang chi tiết sản phẩm

Bước 4: Tìm kiếm các thẻ chứa thông tin về thông số kỹ thuật của đồng hồ và lấy thông tin về.

Get attribute

```
Price = []
name = []
Gender = []
Face_Diameter = []
Glass_material = []
Wire_material = []
Apparatus = []
Waterproof = []
Origin =[]
agency = []

for (i,link) in enumerate(allLinks):
  #try catch để nếu xảy ra lỗi thì vẫn tiếp tục crawl k bị dừng lại
  try:
    print(i,link)
    soup=get_attribute(link)
    div = soup.find('div' ,{'class':'thong-tin-san-pham'})
    if div :
      p = soup.find('p',{'class' : 'price'})
      bdi = p.find('bdi')
      price = bdi.text
      price_clean = bs4.BeautifulSoup(price, 'html.parser').text
      Price.append(price_clean)
      detail = div.find_all('p')
```

```
      for i in detail :
        text_content = i.text
        clean_text = bs4.BeautifulSoup(text_content, 'html.parser').text
        clean_text_lower= clean_text.lower()
        txt = ""
        if len(clean_text.split(':')[1]) > 1:
          txt = clean_text.split(':')[1].replace('\n', '')
        else : txt = "NULL"
        if "thương hiệu:" in clean_text_lower :
          agency.append(txt)
          c1 = 1
        elif "xuất xứ" in clean_text_lower :
          Origin.append(txt)
          c2 =1
        elif "chống nước" in clean_text_lower :
          Waterproof.append(txt)
          c3 =1
        elif "máy:" in clean_text_lower :
          Apparatus.append(txt)
          c4=1
        elif "dây đeo:" in clean_text_lower:
          Wire_material.append(txt)
          c5=1
        elif "kính:" in clean_text_lower and "chân kính" not in clean_text_lower :
          Glass_material.append(txt)
          c6=1
        elif "số hiệu sản phẩm" in clean_text_lower :
          name.append(txt)
          c7=1
        elif "đường kính mặt số" in clean_text_lower :
          Face_Diameter.append(txt)
          c8=1
        elif "giới tính"  in clean_text_lower :
          Gender.append(txt)
          c9=1
```

Hình 06: Code lấy thông tin chi tiết sản phẩm

Bước 5: Lưu dữ liệu thu được vào file csv

```
a = {'name' :name ,'Gender':Gender,'Face_Diameter':Face_Diameter,'Glass_material':Glass_material,'Wire_material':Wire_mater
df = pd.DataFrame.from_dict(a, orient='index')
df = df.transpose()
df.head()
```

	name	Gender	Face_Diameter	Glass_material	Wire_material	Apparatus	Waterproof	Origin	agency	Price
0	MTP-1370D-7A2VDF	Nam	40mm	Mineral Crystal (Kính Cứng)	NULL	Quartz (Pin)	5 ATM	Nhật Bản	Casio	1.607.000 ₫
1	RA-AR0001S10B	Nam	40.8 mm	Sapphire (Kính Chống Trầy)	Thép Không Gỉ	Automatic (Tự Động)	5 ATM	Nhật Bản	Orient	11.760.000 ₫
2	EFV-550L-1AVUDF	Nam	47 mm	Mineral Crystal (Kính Cứng)	Dây Da Chính Hãng	Quartz (Pin)	10 ATM	Nhật Bản	Casio	3.529.000 ₫
3	MTP-1381D-1AVDF	Nam	39.9mm	Mineral Crystal (Kính Cứng)	NULL	Quartz (Pin)	5 ATM	Nhật Bản	Casio	1.710.000 ₫
4	GA-110-1BDR	Nam	53mm	Mineral Crystal (Kính Cứng)	Dây Cao Su	Quartz (Pin)	20 ATM	Nhật Bản	Casio	4.612.000 ₫

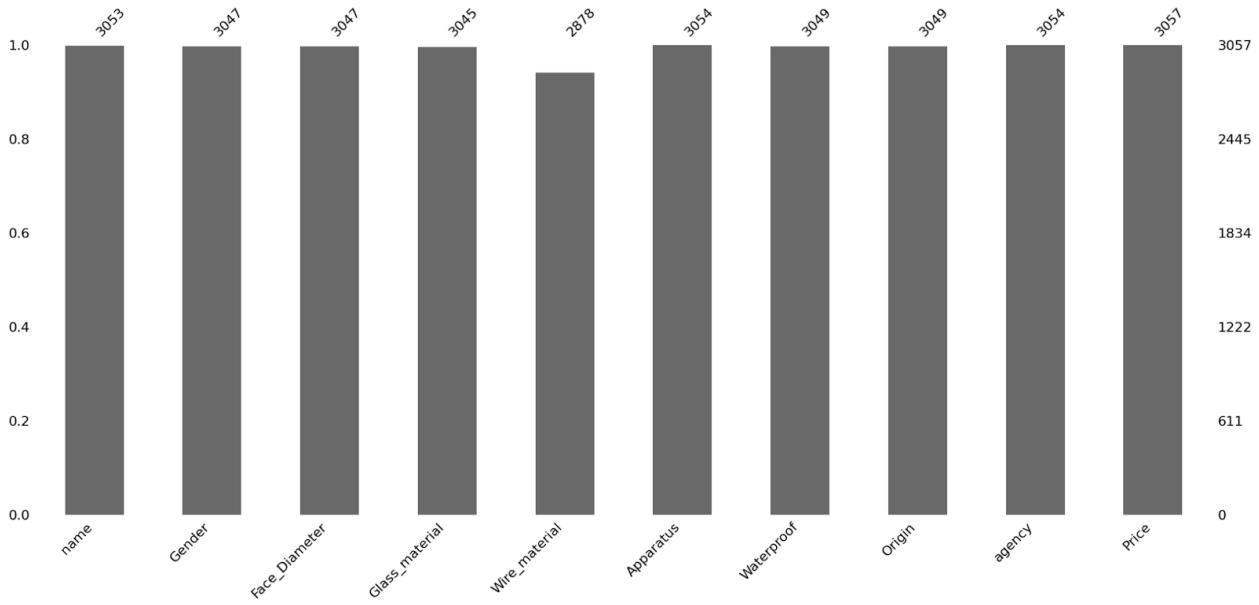
Save CSV

```
dataset = pd.DataFrame(data=df)
dataset.to_csv('dongho_haitrieu.csv', encoding='utf-8-sig', index=False)
```

Hình 07: Lưu dữ liệu vào file csv

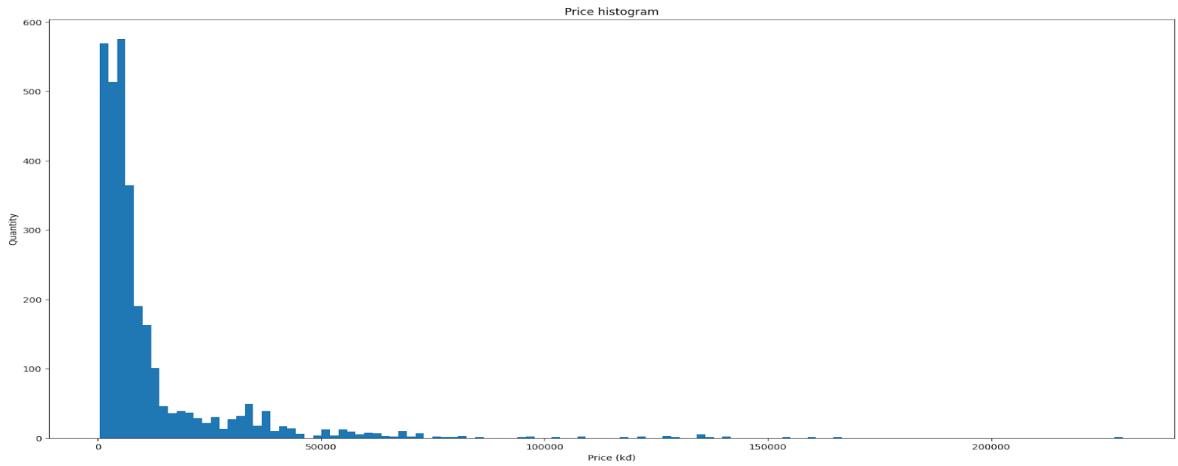
2.2. Mô tả dữ liệu

- Dữ liệu được crawl từ 3 trang web khác nhau, mỗi trang web lại có một cấu trúc khác nhau nên khi crawl sẽ lấy ra các cột dữ liệu cần thiết :
    - Các dữ liệu từ các trang khi crawl xong bao gồm 10 cột: Name, Gender, Face Diameter, Glass Material, Wire Material, Apparatus, Waterproof, Origin, Agency, Price.
    - DaNaWatch: dữ liệu bao gồm 462 mẫu.
    - Đăng Quang Watch: dữ liệu sau crawl bao gồm 1713 mẫu.
    - Đồng hồ Hải Triều: Dữ liệu bao gồm 881 mẫu.
- Gộp các bảng dữ liệu với nhau sau crawl ta được bảng dữ liệu với kích thước: **3057 hàng** và **10 cột**, cụ thể như sau :
    - Số lượng dữ liệu trống của toàn bộ dữ liệu được thể hiện như sau :

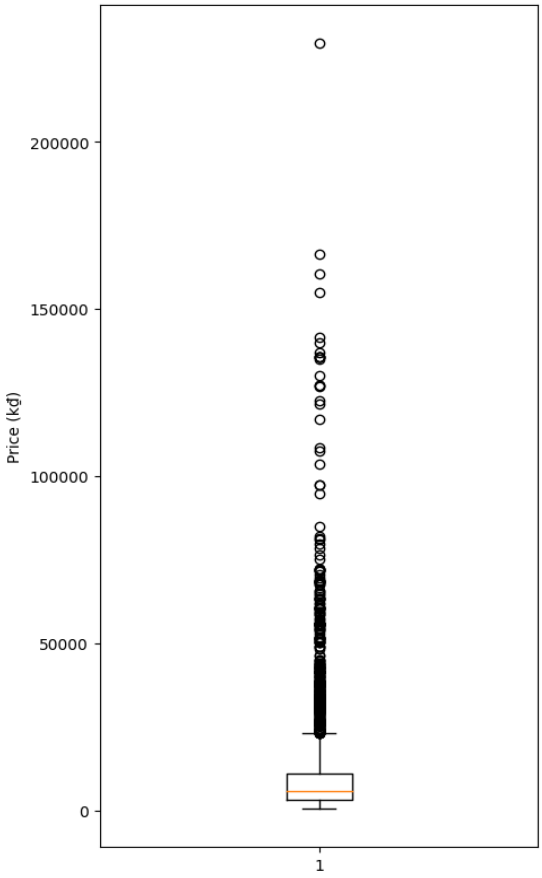


Hình 08: Số lượng dữ liệu trống của toàn bộ dữ liệu thô

- Phân bố dữ liệu của trường dự đoán Price:



Hình 09: Phân bố dữ liệu của trường dự đoán Price



Hình 10: Biểu đồ hộp của cột Price

- Hầu hết các đồng hồ phân bố dưới 100 triệu đồng, có một số mẫu đồng hồ lên đến hơn 100 triệu thậm chí hơn 200 triệu thuộc hãng Doxa và vài đồng hồ khác, loại hàng limited nhưng số lượng không đáng kể. Nên trong phần làm sạch dữ liệu sẽ xóa các đồng hồ đó ra tập dữ liệu.

3. Trích xuất đặc trưng

3.1. Loại bỏ các hàng và các cột dữ liệu không cần thiết

- Toàn bộ dữ liệu thô bao gồm tổng cộng **3057 hàng** với **10 cột**. Trong đó, có các cột không cần thiết cho mô hình dự đoán giá đồng hồ, ta sẽ loại bỏ các cột này. Đó là:
    - *Name*: mã của đồng hồ
    - *Gender* : loại đồng hồ theo giới tính của người dùng
- Mô hình dự đoán giá đồng hồ này được xây dựng để dự đoán giá của các loại đồng hồ, nên ta sẽ loại bỏ cột Gender do giá của đồng hồ không phụ thuộc vào loại đồng hồ dành cho giới tính nào.
- Sau khi loại bỏ các hàng và các cột dữ liệu không cần thiết, bộ dữ liệu còn tổng cộng **3057 hàng** với **8 cột** đặc trưng.

```
Face_Diameter
Glass_material
Wire_material
Apparatus
Waterproof
Origin
agency
Price
dtype: object
```

3.2. Làm sạch dữ liệu với đúng kiểu dữ liệu

- Bộ dữ liệu bao gồm 8 cột với 5 dữ liệu phân loại và 3 dữ liệu dạng số.

a) Dữ liệu phân loại:

- Đối với các dữ liệu phân loại, ta chọn ra các loại phổ biến nhất trong bộ dữ liệu, sau đó chuyển các chuỗi ký tự dài thành các loại tương ứng. Cụ thể:
    - *'Glass_material'* (Chất liệu mặt đồng hồ): Mineral Crystal, Sapphire Crystal, Acrylic Crystal, Resin Glass, Hardened Crystex Crystal, Krysternacrystal, Perspex, Hardlex Crystal, Sapphire, CrystalGlass.
    - *'Wire_material'* (Chất liệu dây đồng hồ): dây kim loại, dây cao su, dây nhựa, dây da, thép không gỉ, thép non, thép không gỉ 316l, thép không gỉ 316l mạ vàng pvd,...
    - *'Apparatus' (*Bộ máy): Quartz, Touch Solar, Automatic, Miyota Japan, Eco-Drive, Hand-Wound.
    - *'Origin' (Xuất xứ):* Nhật Bản, Đức, Thụy Sĩ, Mỹ, Áo, Singapore, Hong Kong, Singapore, Thụy Điển, Đan Mạch, Việt Nam .
    - *'Agency' (Hãng):* Casio, Bentley, Tissot, Wenger, Victorinox, Seiko, Starke, Bulova, Fossil, Orient, Citizen, Reef Tiger.

b) Dữ liệu dạng số

- Đối với các dữ liệu dạng số, ta lấy số tương ứng với đơn vị của từng đặc trưng. Sau đó chuyển thành kiểu dữ liệu số thực. Cụ thể:
  - *'Face_Diameter'*: Kích thước mặt đồng hồ, đơn vị *mm*
  - *'Waterproof':* Độ chống nước, đơn vị *ATM*
- *'Price'*: Giá đồng hồ
- Dữ liệu sau khi làm sạch về đúng kiểu dữ liệu :

	Face_Diameter	Glass_material	Wire_material	Apparatus	Waterproof	Origin	agency	Price
0	45.4	Mineral Crystal	dây kim loại	Quartz	20	Nhật Bản	Casio	5695000
1	45.4	Mineral Crystal	dây kim loại	Quartz	20	Nhật Bản	Casio	5695000
2	45.4	Mineral Crystal	dây cao su	Quartz	20	Nhật Bản	Casio	5515000
3	45.4	Mineral Crystal	dây nhựa	Touch Solar	20	Nhật Bản	Casio	4752000
4	45.4	Mineral Crystal	dây cao su	Quartz	20	Nhật Bản	Casio	5515000

Hình 11: Dữ liệu sau khi trả về đúng kiểu dữ liệu

3.3. Xử lý dữ liệu trống

- Kiểm tra số lượng dữ liệu trống của các cột dữ liệu, ta có cột *'Price'* không có dữ liệu trống.
- Tiến hành thay thế dữ liệu trống cho các cột dữ liệu thành giá trị random trong khoảng giá trị nhất định và cho các cột dữ liệu phân loại như Wire Material… bằng giá trị random trong các loại đã có trong bộ dữ liệu.
- Dữ liệu sau khi xử lý dữ liệu trống:

```
name              4
Gender           10
Face_Diameter    10
Glass_material   12
Wire_material   179
Apparatus         3
Waterproof        8
Origin            8
agency            3
Price             0
dtype: int64
```

1	Face_Diameter	Glass_material	Wire_material	Apparatus	Waterproof	Origin	agency	Price
2	45.4	Mineral Crystal	dây kim loại	Quartz	20	Nhật Bản	Casio	5695000
3	45.4	Mineral Crystal	dây kim loại	Quartz	20	Nhật Bản	Casio	5695000
4	45.4	Mineral Crystal	dây cao su	Quartz	20	Nhật Bản	Casio	5515000
5	45.4	Mineral Crystal	dây nhựa	Touch Solar	20	Nhật Bản	Casio	4752000
6	45.4	Mineral Crystal	dây cao su	Quartz	20	Nhật Bản	Casio	5515000

Hình 12: Dữ liệu sau khi xử lý dữ liệu trống

3.4. Chuyển các dữ liệu phân loại thành dữ liệu dạng số

- Sử dụng *LabelEncoder* của thư viện *sklearn*, chuyển các cột dữ liệu *'Glass_material', 'Wire_material', 'Apparatus', 'Origin', 'agency'* thành dạng số. Dữ liệu sau khi chuyển thành dạng số:

	Face_Diameter	Glass_material	Wire_material	Apparatus	Waterproof	Origin	agency	Price
0	45.4	8	12	7	20	10	4	5695000
1	45.4	8	12	7	20	10	4	5695000
2	45.4	8	8	7	20	10	4	5515000
3	45.4	8	15	9	20	10	4	4752000
4	45.4	8	8	7	20	10	4	5515000

Hình 13: Dữ liệu sau khi chuyển dữ liệu phân loại thành dạng số

3.5. Chuẩn hóa dữ liệu

- Chuẩn hóa dữ liệu sử dụng *StandardScaler* của thư viện *sklearn* với thuộc tính *fit_transform* cho tập dữ liệu huấn luyện và *transform* cho tập dữ liệu kiểm thử.
- Chuẩn hóa Z-score (Standardization): Phương pháp này biến đổi dữ liệu sao cho có giá trị trung bình bằng 0 và độ lệch chuẩn bằng 1. Công thức chuẩn hóa Z-score cho một giá trị là:

$$x' = x - mean(X) / std(X)$$

trong đó :

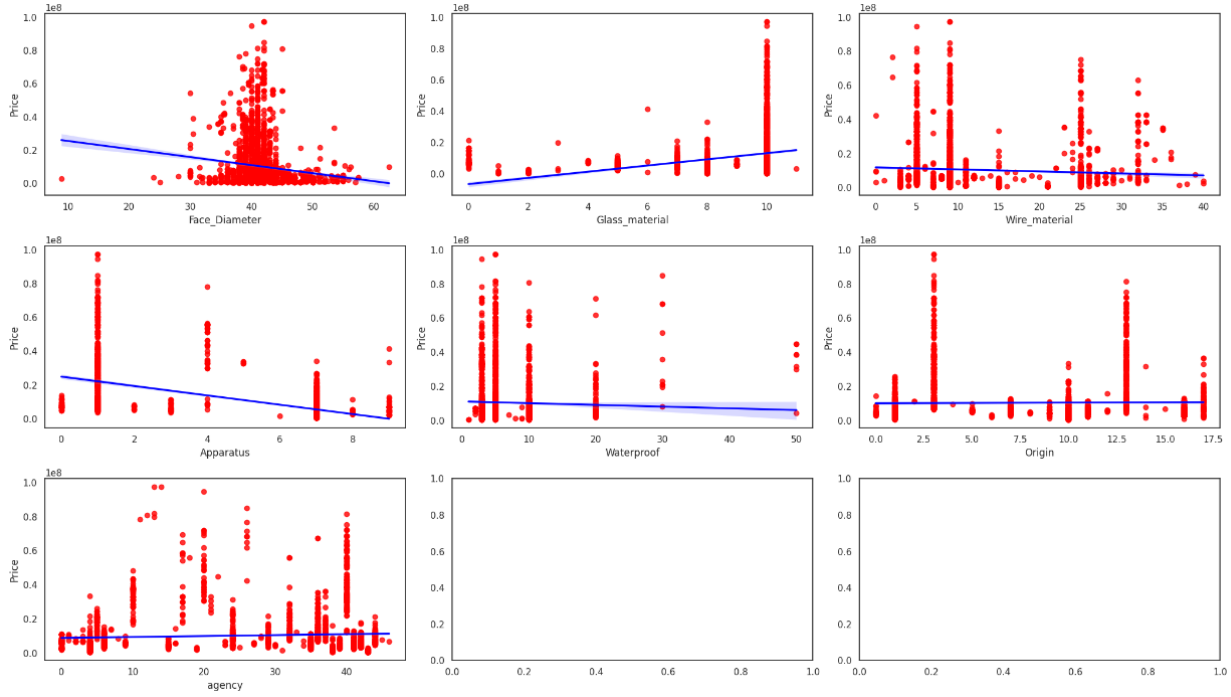
+ x là giá trị ban đầu
+ x’ là giá trị đã được chuẩn hóa
+ mean(X) là giá trị trung bình của tập dữ liệu
+ std(X) là độ lệch chuẩn của tập dữ liệu

	Face_Diameter	Glass_material	Wire_material	Apparatus	Waterproof	Origin	agency
489	-0.765112	0.722829	1.420744	0.646413	-0.444261	-0.436547	0.875667
1547	1.440685	-0.261749	-0.753728	-1.543627	0.509901	0.286717	0.717320
2262	-0.458752	0.722829	-0.995336	-1.543627	-0.825926	-1.159811	-0.285542
2508	-0.152391	-0.261749	-0.512120	0.646413	-0.444261	-1.521444	-1.499534
255	0.000789	-0.261749	-0.270512	-1.543627	-0.825926	0.286717	-0.074413
...	...	...	...	...	...	...	...
1739	-1.071473	0.722829	1.420744	0.646413	-0.444261	0.467534	0.506191
2715	-0.458752	0.722829	-0.512120	-1.543627	-0.444261	-1.159811	-0.496671
2204	-0.918293	0.722829	-0.995336	-1.543627	-0.444261	-1.521444	-0.074413
2432	-0.458752	0.722829	-0.995336	-1.543627	0.509901	-1.159811	-1.182840
2693	1.348777	-0.261749	-0.995336	-1.908634	0.509901	-1.521444	-1.393969

Hình 14: Dữ liệu huấn luyện sau khi chuẩn hóa dữ liệu

3.6. Độ tương quan giữa các đặc trưng với mục tiêu

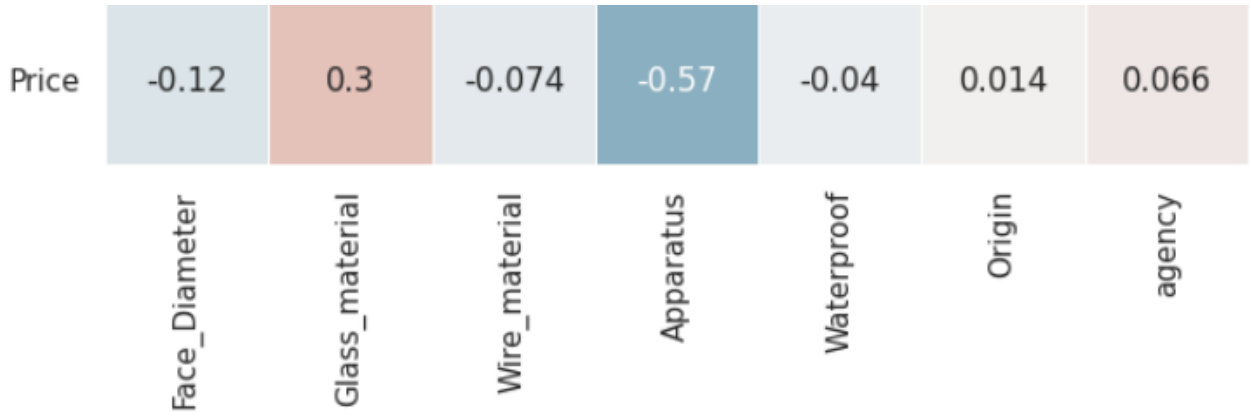
- Tiến hành trực quan hóa sự tương quan của các đặc trưng dạng số so với đặc trưng mục tiêu là *'Price'* bằng sơ đồ *regplot*. Ta thu được sơ đồ dưới đây:



Hình 15: Sự tương quan giữa các đặc trưng dạng số với đặc trưng mục tiêu

- Các đặc trưng *'Face_Diameter'*, *'Glass_material'* và *'Apparatus'* có sự tương quan nhẹ với giá đồng hồ, trong khi *'Face_Diameter'* và *'Apparatus'* có tương quan âm với giá thì *'Glass_material'* lại có tương quan dương. Còn đặc trưng

'*Wire_material*', '*Waterproof*', '*Origin*', '*agency*' có độ tương quan không đáng kể.



Hình 16: Sự tương quan giữa tất cả đặc trưng với đặc trưng mục tiêu

- Nhìn vào bảng so sánh thì ta thấy 'Apparatus' có độ tương quan cao nhất so với 'Price', ngoài ra các đặc trưng khác cũng có độ tương quan nhẹ không đáng kể. Vì vậy ta sẽ lựa chọn các đặc trưng này cho mô hình dự đoán. Có tổng cộng 7 đặc trưng cho mô hình đó là *'Face_Diameter', 'Glass_material', 'Wire_material', 'Apparatus', 'Waterproof', 'Origin', 'agency'*.

4. Mô hình hóa dữ liệu

Để đáp ứng cho yêu cầu bài toán Dự đoán giá đồng hồ, nhóm đã chọn ra 2 mô hình cho để giải quyết bài toán này: Hồi quy của thư viện XGBoost (XGBoost Regression) và Random Forest Regression.

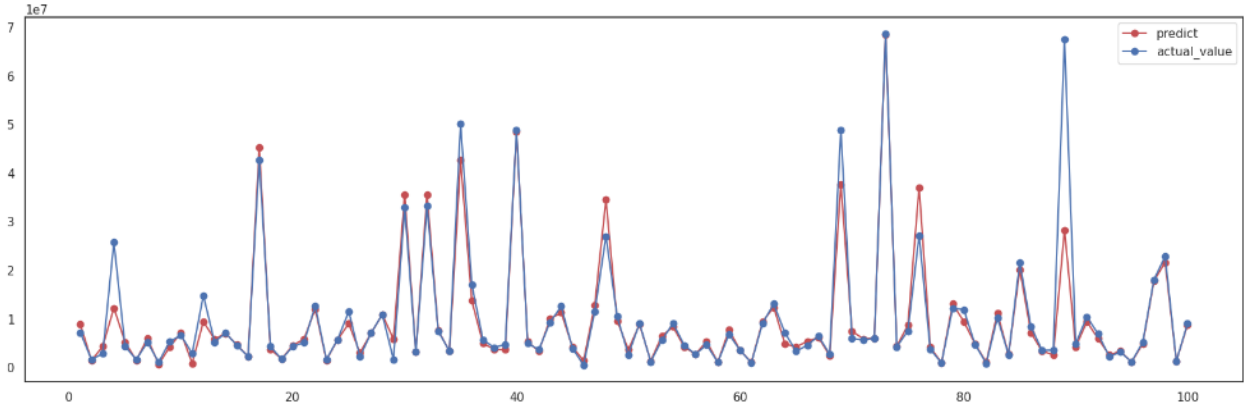
4.1. Random Forest Regression

- **Cơ sở lý thuyết**: Random Forest (Rừng ngẫu nhiên) là một phương pháp học máy dựa trên kỹ thuật Ensemble Learning, được xây dựng dựa trên các cây quyết định độc lập. Ý tưởng chính của Random Forest là kết hợp dự đoán của nhiều cây quyết định (decision trees) để tạo ra một dự đoán cuối cùng. Dưới đây là cơ lý thuyết của Random Forest:
    - Tạo ra tập hợp cây quyết định: Random Forest bắt đầu bằng việc tạo ra một tập hợp các cây quyết định. Số lượng cây trong tập hợp được xác định trước và là một tham số đầu vào.
    - Bootstrap sampling: Mỗi cây quyết định trong tập hợp được huấn luyện trên một tập con dữ liệu huấn luyện được lấy mẫu ngẫu nhiên từ tập dữ liệu huấn

luyện ban đầu. Quá trình này được gọi là bootstrap sampling

- Xây dựng cây quyết định: Mỗi cây quyết định trong tập hợp được xây dựng bằng cách chia tập con dữ liệu huấn luyện dựa trên các đặc trưng và giá trị đặc trưng tốt nhất để giảm thiểu hàm mất mát (loss function). Quá trình chia dữ liệu tiếp tục cho đến khi đạt được một điều kiện dừng, chẳng hạn như đạt đến một độ sâu tối đa hoặc số lượng mẫu nhỏ nhất trong mỗi nút lá.
- Dự đoán bằng cách kết hợp các cây: Khi có dữ liệu mới, mỗi cây trong tập hợp được sử dụng để dự đoán giá trị đầu ra. Đối với bài toán hồi quy, dự đoán cuối cùng được tính bằng cách lấy trung bình các dự đoán của các cây quyết định.

- **Bộ tham số của mô hình**:
  - n_estimators: Đây là tham số quan trọng nhất và xác định số lượng cây quyết định trong tập hợp. Giá trị của tham số này cần được lựa chọn một cách cân nhắc, vì việc tăng số lượng cây có thể làm tăng độ phức tạp và thời gian huấn luyện.
  - random_state: Tham số này định rõ hạt giống (seed) cho việc tạo ra các số ngẫu nhiên. Điều này đảm bảo rằng mô hình sẽ có cùng kết quả khi được huấn luyện lại trên cùng một tập dữ liệu.
- **Chia tập dữ liệu**:
  - Chia tập dữ liệu Huấn luyện và Kiểm thử : 80% cho huấn luyện và 20% cho kiểm thử.
- **Biểu đồ thể hiện giá dự đoán và giá thực tế:**



Hình 17: Sự chênh lệch giá cả dự đoán so với thực tế Random Forest Regression

- **Kết quả**: Nhìn trên đồ thị ta thấy đường màu đỏ và màu xanh còn nhiều chỗ chênh lệch khá nhiều. Kết quả dự đoán đánh giá của mô hình của mô hình có độ chính xác đạt **88.9%** trên tập dữ liệu kiểm thử.
- **Đánh giá**: Độ hiệu quả dự đoán với các metrics, các metrics sử dụng để đánh giá bao gồm: *RMSE*, *MAE* và *R2.*

	RMSE *(nghìn đồng)*	**MAE** *(nghìn đồng)*	**R2** *(%)*
Số liệu	4151184.705136784	1719891.8763456596	88.9

Bảng 01: Bảng giá trị các metrics của mô hình Random Forest Regression

4.2. Mô hình Hồi quy của thư viện XGBoost - XGBoost Regression

- **Cơ sở lý thuyết**: Mô hình XGBoost được xây dựng bằng cách sử dụng các cây quyết định nhị phân (binary decision trees) dưới dạng các mô hình học tập yếu. Các cây quyết định này được xây dựng tuần tự, mỗi cây được xây dựng để giảm hàm mất mát của mô hình trước đó. XGBoost sử dụng gradient descent để tìm kiếm hướng điều chỉnh các cây quyết định, từ đó cải thiện mô hình dự đoán.
- Các đặc điểm chính của XGBoost bao gồm:
    - Regularization: XGBoost có thể áp dụng regularization để tránh overfitting và tăng tính tổng quát của mô hình.
    - Hàm mất mát tùy chỉnh: Người dùng có thể định nghĩa các hàm mất mát tùy chỉnh dựa trên yêu cầu của bài toán cụ thể.
    - Xử lý dữ liệu thiếu: XGBoost có thể tự động xử lý các giá trị thiếu trong dữ liệu đầu vào mà không cần tiền xử lý bổ sung.
    - Tăng tốc: XGBoost sử dụng một số kỹ thuật tối ưu hóa để tăng tốc quá trình huấn luyện, bao gồm sự song song hóa và gian lận thuật toán.
    - Xác định độ quan trọng của đặc trưng: XGBoost có thể đánh giá độ quan trọng của các đặc trưng trong mô hình, giúp xác định các đặc trưng quan trọng nhất đối với dự đoán.
- **Tìm kiếm tham số tốt nhất**: Để tìm siêu tham số tối ưu cho mô hình XGBoost,

sử dụng RandomizedSearchCV từ thư viện scikit-learn. RandomizedSearchCV giúp tìm kiếm siêu tham số tốt nhất trong một phạm vi giá trị được chỉ định.

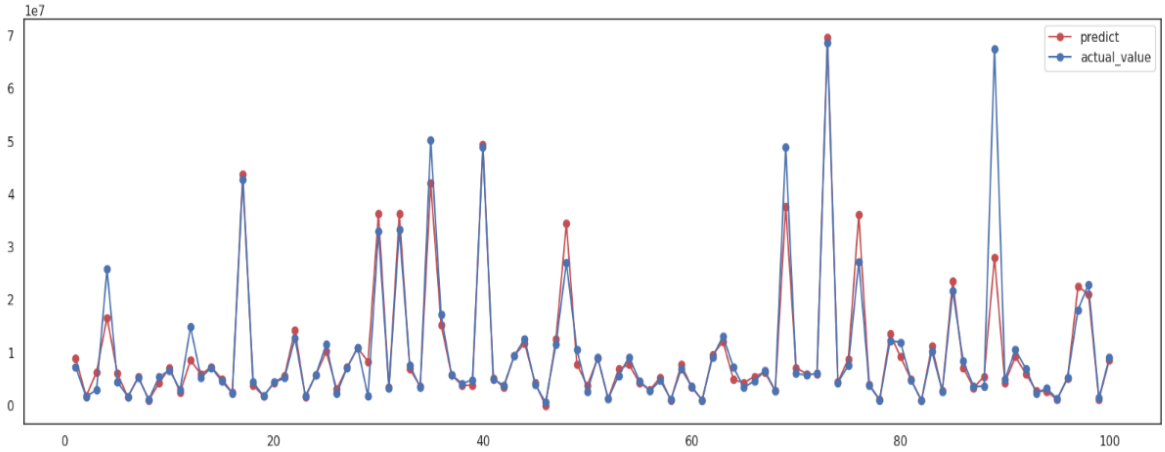
- n_estimators :Số estimators trong mô hình tìm kiếm
- max_depth :Độ sâu tìm kiếm tối đa
- booster :Mô hình để chạy
- learning_rate :Tốc độ học
- min_child_weight :Trọng số tối thiểu của giá trị con
- base_scrore :Điểm khởi tạo tìm tham số

- **Chia tập dữ liệu**:
  - Chia tập dữ liệu Huấn luyện và Kiểm thử : 80% cho huấn luyện và 20% cho kiểm thử
- **Bộ tham số của mô hình**: Sau khi sử dụng RandomizedSearchCV để tìm được bộ tham số tốt nhất thì ta được bộ tham số như sau :

Tham số	Giá trị
n_estimators	300
max_depth	5
booster	gbtree
learning_rate	0.3
min_child_weight	1
base_scrore	0.25

Bảng 02: Bảng tham số của mô hình

- **Biểu đồ thể hiện giá dự đoán và giá thực tế:**



Hình 18:Sự chênh lệch giá cả dự đoán so với thực tế XGBoost Regression

- **Kết quả**: Nhìn trên đồ thị ta thấy đường màu đỏ và màu xanh có vài chỗ chênh lệch khá nhiều còn lại thì chênh lệch rất ít . Kết quả dự đoán đánh giá trên hàm score có sẵn của mô hình của mô hình có độ chính xác đạt **90.93%** trên tập dữ liệu kiểm thử.
- **Đánh giá**: Độ hiệu quả dự đoán với các metrics, các metrics sử dụng để đánh giá bao gồm: *RMSE*, *MAE* và *R2*.

Metrics	**RMSE** *(nghìn đồng)*	**MAE** *(nghìn đồng)*	**R2** *(%)*
Số liệu	3768503.09450	1727134.33310	90.93

Bảng 03: Bảng giá trị các metrics của mô hình XGBoost Regression

4.3. So sánh hiệu quả của các mô hình

	RMSE *(nghìn đồng)*	**MAE** *(nghìn đồng)*	**R2** *(%)*
RandomForest	4151184.70514	1719891.87635	88.9
XGBoost	3768503.09450	1727134.33310	90.93

Bảng 04: Bảng số liệu của các metrics giữa 2 mô hình

- **Chú thích :**
  - R2 : Là phép đo dùng để đo lường tỷ lệ phương sai của biến phụ thuộc mà mô hình có thể giải thích được so với tổng phương sai của biến phụ thuộc. Giá trị R2 nằm trong khoảng từ 0 đến 1, và càng gần 1 thì mô hình giải thích dữ liệu tốt hơn.

- RMSE : Là phép đo dùng để đo lường độ lớn sai số dự đoán giữa giá trị dự đoán của mô hình và giá trị thực tế trong dữ liệu. Nó tính căn bậc hai của trung bình bình phương của sai số. Giá trị RMSE càng nhỏ, mô hình càng có khả năng dự đoán chính xác và gần với giá trị thực tế.
- MAE : MAE tính trung bình giá trị tuyệt đối của sai số giữa giá trị dự đoán và giá trị thực tế. MAE càng nhỏ thì mô hình dự đoán tốt hơn.

- **Đánh giá:**
  - Thông số MAE của thuật toán RandomForest Regression nhỏ hơn XGBooots Regression, điều đó cho thấy độ sai lệch các giá trị của thuật toán RandomForest Regression ít hơn.
  - Hiệu suất R2 của XGBoost lại cao hơn một ít và RMSE thấp hơn cho thấy mô hình XGBoost dự đoán tốt hơn.

➔ Dựa vào một số đánh giá trên ,ta thấy được XGBoost Regression cho số liệu đánh giá tốt hơn, vì mô hình tìm được bộ tham số hiệu quả, bản thân thư viện XGBoost nói chung là thư viện được tối ưu hóa mạnh nên độ hiệu quả được các metric đánh giá cho kết quả cao, đáng tin cậy.

5. Dự đoán và kết luận

5.1. Dự đoán

- Load các file đã được lưu trong quá trình làm :

```
import pickle
with open('XGB_model.pkl', 'rb') as file:
    xgb_model = pickle.load(file)
with open('encoder.pkl', 'rb') as file:
    le_dict = pickle.load(file)
with open('scaler.pkl', 'rb') as file:
    number_normal_scaler = pickle.load(file)
```

Trong đó :

- XGB_model.pkl : File mode được lưu sao khi train
- encoder.pkl : Lưu file sau khi fit_transform() sử dụng labelEncoder
- scaler.pkl : Lưu file sau khi fit_transform() sử dụng StandardScaler

- Khởi tạo đối tượng muốn dự đoán :

```
watch = {

'Glass_material' : ['Mineral Crystal']   ,
 'agency' : ['Citizen'] ,
 'Origin' : ['Nhật Bản']  ,
 'Waterproof' :[5]   ,
 'Wire_material'     : ['thép không gỉ'] ,
'Face_Diameter'    :[39 ],
'Apparatus'       :['Quartz']

}
```

- Thực hiện các bước xử lý dữ liệu trước khi dự đoán :

```
import pandas as pd
vars_normalizing =   ['Face_Diameter','Glass_material', 'Wire_material', 'Apparatus',  'Waterproof', 'Origin', 'agency']
def predict(watch) :
  input_data = pd.DataFrame(watch)
    # label encoder
  input_data['Wire_material'] = le_dict['Wire_material'].transform(input_data['Wire_material'])
  input_data['Glass_material'] = le_dict['Glass_material'].transform(input_data['Glass_material'])
  input_data['Apparatus'] = le_dict['Apparatus'].transform(input_data['Apparatus'])
  input_data['Origin'] = le_dict['Origin'].transform(input_data['Origin'])
  input_data['agency'] = le_dict['agency'].transform(input_data['agency'])
    #tranform


  input_data[vars_normalizing] = number_normal_scaler.transform(input_data[vars_normalizing])
  input_data = input_data.reindex(columns=vars_normalizing)
  input_data = input_data[vars_normalizing]

  # predict
  price = xgb_model.predict(input_data)
  return int(price)
```

- Kết quả đạt được :

```
print("Giá đồng hồ dự đoán được :" , predict(watch))
```

```
Giá đồng hồ dự đoán được : 3562557
```

5.2. Kết luận

5.2.1. Thu nhập dữ liệu

Vì số lượng đồng hồ trên thị trường còn có rất nhiều nên ta có thể thu nhập thêm dữ liệu để tăng kích thước của bộ dữ liệu để có được độ chính xác cao hơn. Tổng kích thước của toàn bộ dữ liệu thô hiện có là 3057 hàng và 10 cột. Dữ liệu được thu thập chứa nhiều giá trị không cần thiết và chênh lệch rất cao, cần xử lý trả về đúng kiểu dữ liệu và loại bỏ bớt các dữ liệu.

5.2.1. Thống kê mô tả trực quan về dữ liệu

Dữ liệu sau khi Crawl được thống kê, mô tả một các trực quan bằng các thông số, biểu đồ cột, biểu đồ thể hiện phân bố của Price. Những dữ liệu trống được liệt kê chi tiết, các đồng hồ có giá rất cao và limited sẽ được làm sạch sau đó.

5.2.2. Làm sạch dữ liệu

Dữ liệu được làm sạch xử lý trả về đúng kiểu dữ liệu phù hợp với từng đặc trưng. Nhờ loại bỏ các dữ liệu không cần thiết, xử lý các đặc trưng có số lượng dữ liệu trống quá lớn, các đặc trưng được sử dụng trong mô hình dự đoán có độ tương quan với đặc trưng mục tiêu cao, giúp cho hiệu quả mô hình dự đoán được cải thiện. Xử lý dữ liệu trống hợp lý cho loại đồng hồ, với đặc trưng agency, Apparatus… được thay thế phù hợp, logic, tránh ảnh hưởng kết quả dự đoán vì giá đồng hồ có giá cao hơn nhiều. Tổng kích thước của toàn bộ dữ liệu sau khi xử lý và dùng để dự đoán là ***3035 hàng với 8 cột***. Sau khi tính toán độ tương quan của toàn bộ đặc trưng và thử nghiệm mô hình dự đoán cho từng đặc trưng, đã đưa ra được các đặc trưng cần thiết cho mô hình.

5.2.3. Mô hình hóa dữ liệu

Đã xây dựng được 2 mô hình dự đoán giá đồng hồ là Random Forest Regression và XGBoost Regression với độ chính xác được đánh giá cao. Qua đó cho ta thấy được mô hình XGBoost Regression sẽ phù hợp với bài toán đề ra hơn. Tuy nhiên chúng ta cần thu nhập thêm dữ liệu và tìm kiếm bộ tham số tốt nhất cho 2 mô hình để có thể cải thiện độ chính xác của mô hình cho ra là cao hơn so với hiện tại.