# Yêu cầu

Viết cơ sở toán học, mã giã và cài đặt 1 trong các thuật toán sau (về giải phương trình hồi quy phi tuyến):

- Phương pháp Gradient Descent (Tối ưu hóa dựa trên đạo hàm)

- Phương pháp Gauss-Newton

- Phương pháp Spline hoặc Spline Cubic

- Phương pháp hồi quy quá trình Gaussian (Gaussian Process)

- Phương pháp tối ưu hóa tổng quát (Global Optimization Techniques)

- Nhóm thuật toán AI: Neural Networks, SVM, Mô hình cây quyết định và các phương pháp kết hợp (Ensemble Methods)

# Mục Lục

[Yêu cầu 1](#_Toc195834835)

[Mục Lục 1](#_Toc195834836)

[Bảng phân công công việc 2](#_Toc195834837)

[1. Cơ sở toán học của thuật toán Gradient Descent 2](#_Toc195834838)

[1.1. Tối ưu phi tuyến 2](#_Toc195834839)

[1.2. Phương pháp gradient 3](#_Toc195834840)

[1.3. Phương pháp steepest‐descent 3](#_Toc195834841)

[2. Mã giả 4](#_Toc195834842)

[So sánh Mã giả với ChatGPT 4](#_Toc195834843)

[3. Cài đặt thuật toán trên python 5](#_Toc195834844)

[3.1. Chọn tập dữ liệu về Việt Nam 5](#_Toc195834845)

[3.2. Thực hiện code 6](#_Toc195834846)

[3.3. So sánh Code với ChatGPT 8](#_Toc195834847)

[Tài liệu tham khảo 8](#_Toc195834848)

# Bảng phân công công việc

|  |  |  |
| --- | --- | --- |
| **Thành viên** | | **Công việc phân công** |
| **Mã học viên** | **Họ và Tên** |
| 240104026 | Huỳnh Quốc Bữu | Viết mã giả, cài đặt thuật toán và viết so sánh với ChatGPT |
| 240104039 | Nguyễn Đức Linh | Viết cơ sở toán học |

# Cơ sở toán học của thuật toán Gradient Descent

## Tối ưu phi tuyến

Ta giả sử có mô hình phi tuyến:

Trong nhiều bài toán học máy, đặc biệt là hồi quy phi tuyến (nonlinear regression), mục tiêu là tìm bộ tham số tối ưu để mô hình , khớp tốt nhất với dữ liệu huấn luyện . Điều này thường được thực hiện bằng cách tối thiểu hóa một hàm mất mát, chẳng hạn như tổng bình phương sai số [1]:

Bài toán lúc này là cần tối ưu vô điều kiện tổng quát

Với

,

Điều kiện cần cho điểm cực tiểu cục bộ là

Điều kiện đủ nếu tại ta có

(ma trận Hessian dương xác định, thì là cực tiểu địa phương duy nhất.[2])

## Phương pháp gradient

Giới hạn đầu tiên của khai triển Taylor cho một bước lặp là

Để với một số cần:

Ta gọi là hướng giảm (descent direction) nếu

## Phương pháp steepest‐descent

Steepest‐descent (giảm dốc mạnh nhất) định nghĩa hướng giảm sao cho nó làm giảm nhanh nhất, dưới ràng buộc rằng có độ dài đơn vị theo một chuẩn nhất định[3]. Với norm chuẩn 2, ta có

Trong thực tế người ta bỏ điều kiện và chọn thẳng [4]

Sau đó, tìm bước (qua line search) sao cho

Khi điều kiện trên được thỏa mãn, ta tiếp tục cập nhật điểm ​ theo công thức:

Sau đó, quá trình lặp tiếp tục cho tới khi hoặc đạt số lần lặp

# Mã giả

**Input:**

x ←

for k = 0 to do

g ←

if then

return x

end if

h ←

α ← LineSearch(x, h)

x ←

end for

**return** x

## So sánh Mã giả với ChatGPT

Nhận xét: Chat GPT nhầm lẫn giữa mã giả và code thực sự bằng python.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer program

AI-generated content may be incorrect.

# Cài đặt thuật toán trên python

## Chọn tập dữ liệu về Việt Nam

1. Chọn tập dữ liệu về giá nhà tại Việt Nam để phân tích
2. Nguồn dữ liệu: <https://www.kaggle.com/datasets/nguyentiennhan/vietnam-housing-dataset-2024/data> [5]
3. File dữ liệu: Github-Code-Team11/datasets/vietnam\_housing\_dataset.csv
4. Giới thiệu tập dữ liệu:

Bộ dữ liệu này chứa thông tin về các bất động sản nhà ở tại Việt Nam. Nó bao gồm các đặc điểm chi tiết của từng bất động sản như vị trí, đặc điểm vật lý, tình trạng pháp lý và nội thất, cùng với giá bán. Bộ dữ liệu này rất hữu ích cho việc phân tích bất động sản, xây dựng mô hình dự đoán giá và phân tích xu hướng thị trường.

Nguồn thu thập dữ liệu: Dữ liệu được thu thập (crawled) từ trang web batdongsan.vn.

1. Mô tả các cột dữ liệu:

* Address (Địa chỉ): Địa chỉ đầy đủ của bất động sản, bao gồm tên dự án (nếu có), đường, phường, quận và thành phố.
* Area (Diện tích): Tổng diện tích của bất động sản, đơn vị tính là mét vuông (m²).
* Frontage (Mặt tiền): Chiều rộng mặt trước của bất động sản, đơn vị tính là mét (m).
* Access Road (Đường vào): Chiều rộng của con đường dẫn vào bất động sản, đơn vị tính là mét (m).
* House Direction (Hướng nhà): Hướng chính mà mặt trước của căn nhà quay về (ví dụ: Đông, Tây, Nam, Bắc).
* Balcony Direction (Hướng ban công): Hướng mà ban công quay về.
* Floors (Số tầng): Tổng số tầng của căn nhà.
* Bedrooms (Số phòng ngủ): Số lượng phòng ngủ trong bất động sản.
* Bathrooms (Số phòng tắm): Số lượng phòng tắm trong bất động sản.
* Legal Status (Tình trạng pháp lý): Trạng thái pháp lý của bất động sản, ví dụ như có sổ hồng, đang trong hợp đồng mua bán, v.v.
* Furniture State (Tình trạng nội thất): Tình trạng trang bị nội thất, ví dụ như đầy đủ nội thất, nội thất cơ bản hoặc chưa có nội thất.
* Price (Giá bán): Giá của bất động sản, đơn vị tính là **tỷ đồng Việt Nam (VND)**.

## Thực hiện code

|  |
| --- |
| import pandas as pd  import numpy as np  # 1) Đọc dữ liệu  CSV\_PATH = "C:\Users\ducli\Downloads\archive (1)/vietnam\_housing\_dataset.csv"  df = pd.read\_csv(CSV\_PATH)  # 2) Giữ lại 3 đặc trưng + đích  features = ["Area", "Bedrooms", "Floors"]  target   = "Price"  data = df[features + [target]].dropna()        # bỏ hàng có NaN  # 3) Chuẩn hoá đặc trưng (z‑score) cho tốc độ hội tụ  X\_raw = data[features].values  y     = data[target].values.reshape(-1, 1)  mu    = X\_raw.mean(axis=0)  sigma = X\_raw.std(axis=0)  Xn    = (X\_raw - mu) / sigma                  # X chuẩn hoá  m     = Xn.shape[0]  # 4) Thêm cột bias  X = np.hstack([np.ones((m, 1)), Xn])          # (m × 4)  theta = np.zeros((X.shape[1], 1))             # khởi tạo 0  # 5) Các hàm tiện ích  def compute\_cost(X, y, theta):      m = len(y)      return (1/(2\*m)) \* np.sum((X @ theta - y) \*\* 2)  def gradient\_descent(X, y, theta, alpha=0.01, iters=1000):      cost\_history = []      m = len(y)      for \_ in range(iters):          error  = X @ theta - y          grad   = (1/m) \* (X.T @ error)          theta -= alpha \* grad          cost\_history.append(compute\_cost(X, y, theta))      return theta, cost\_history  # 6) Huấn luyện  theta\_opt, J\_hist = gradient\_descent(X, y, theta,                                       alpha     = 0.01,                                       iters     = 1000)  # 7) In kết quả  print("Thông số đã học (theta):")  print(f"θ0 (bias)      = {theta\_opt[0,0]:.4f}")  print(f"θ1 (Area)      = {theta\_opt[1,0]:.4f}")  print(f"θ2 (Bedrooms)  = {theta\_opt[2,0]:.4f}")  print(f"θ3 (Floors)    = {theta\_opt[3,0]:.4f}")  print(f"J cuối cùng     = {J\_hist[-1]:.4f}") |

Kết quả thực thi:

A screenshot of a computer program

AI-generated content may be incorrect.

## So sánh Code với ChatGPT

A computer screen with white text

AI-generated content may be incorrect.

Về cơ bản, cả ChatGPT và nhóm đều tính gradient và cập nhật tham số theo cùng một công thức vectorized; độ phức tạp tính toán giống nhau (O(n·d) mỗi vòng, với n số mẫu, d số chiều).

Điểm khác biệt nhỏ: ChatGPT gọi hàm compute\_loss mỗi vòng; cách này thì chia nhỏ các hàm dễ bảo trì ngược lại nếu xét về tối ưu trong tính toán thì cần gọi đến hàm tính toán, sẽ phí ít thời gian so với code inline.

# Tài liệu tham khảo

[1] *Pattern Recognition and Machine Learning*. Accessed: Apr. 18, 2025. [Online]. Available: https://link.springer.com/book/9780387310732

[2] S. P. Boyd and L. Vandenberghe, *Convex optimization*, Version 29. Cambridge New York Melbourne New Delhi Singapore: Cambridge University Press, 2023.

[3] J. Nocedal and S. J. Wright, *Numerical optimization*, Second edition. in Springer series in operations research and financial engineering. New York, NY: Springer, 2006.

[4] W. Hao, “A gradient descent method for solving a system of nonlinear equations,” *Appl. Math. Lett.*, vol. 112, p. 106739, Feb. 2021, doi: 10.1016/j.aml.2020.106739.

[5] “House Price Prediction Dataset Vietnam - 2024.” Accessed: Apr. 18, 2025. [Online]. Available: https://www.kaggle.com/datasets/nguyentiennhan/vietnam-housing-dataset-2024