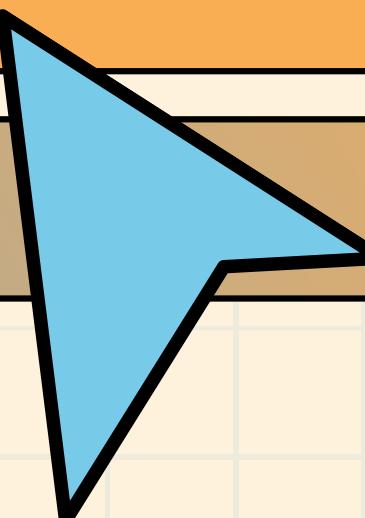


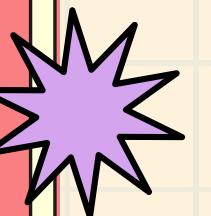
Ứng dụng Linear Regression trong game

Dự đoán mức rank của người chơi

Liên Minh Huyền Thoại

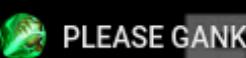


Giới thiệu thành viên

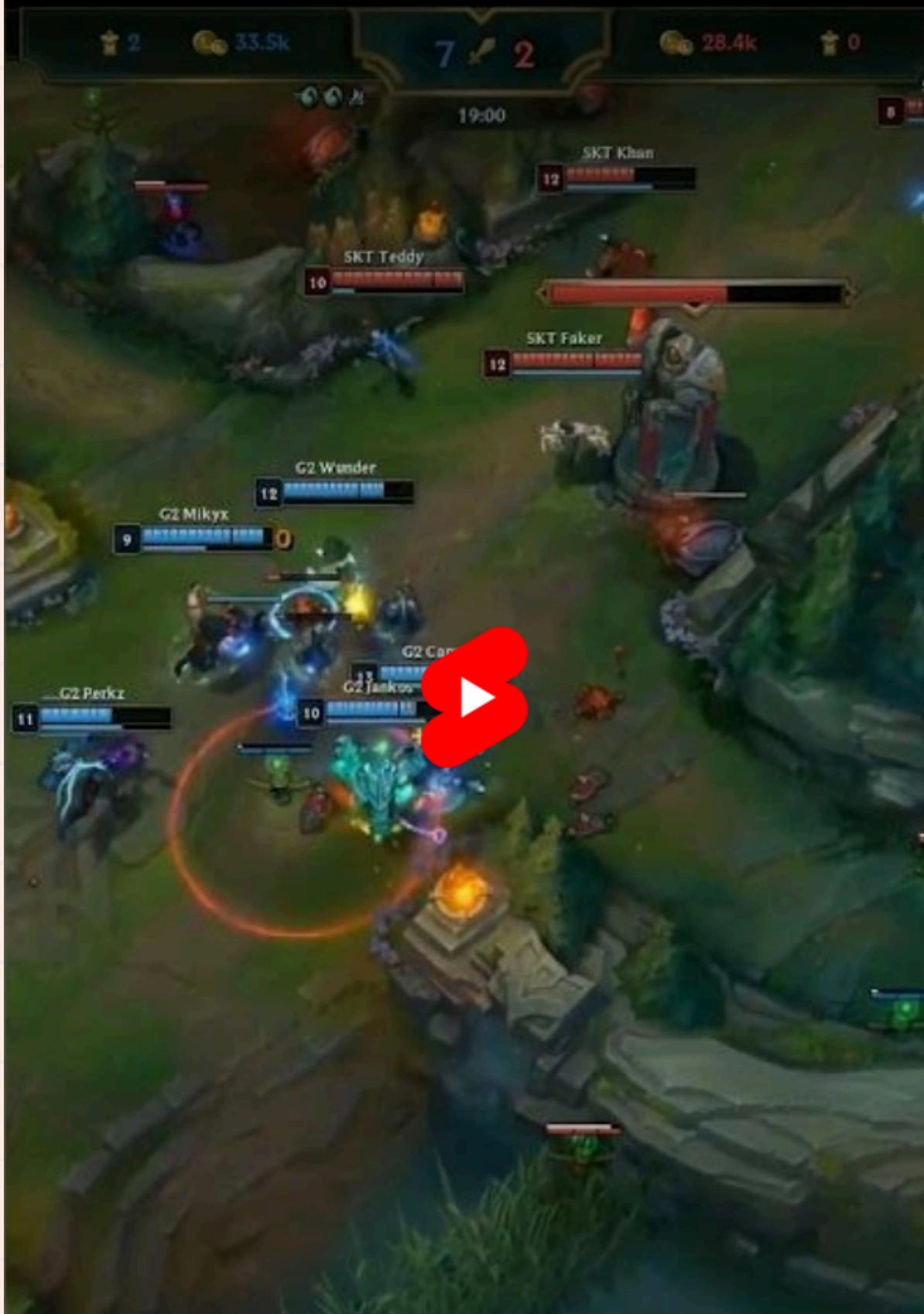


MSSV	Họ và tên
HE191387	Tạ Bảo Ngọc
HE191422	Lê Minh Hoàng
HE194099	Nguyễn Đức Hoàng Phúc
HE191708	Vũ Hải Đăng





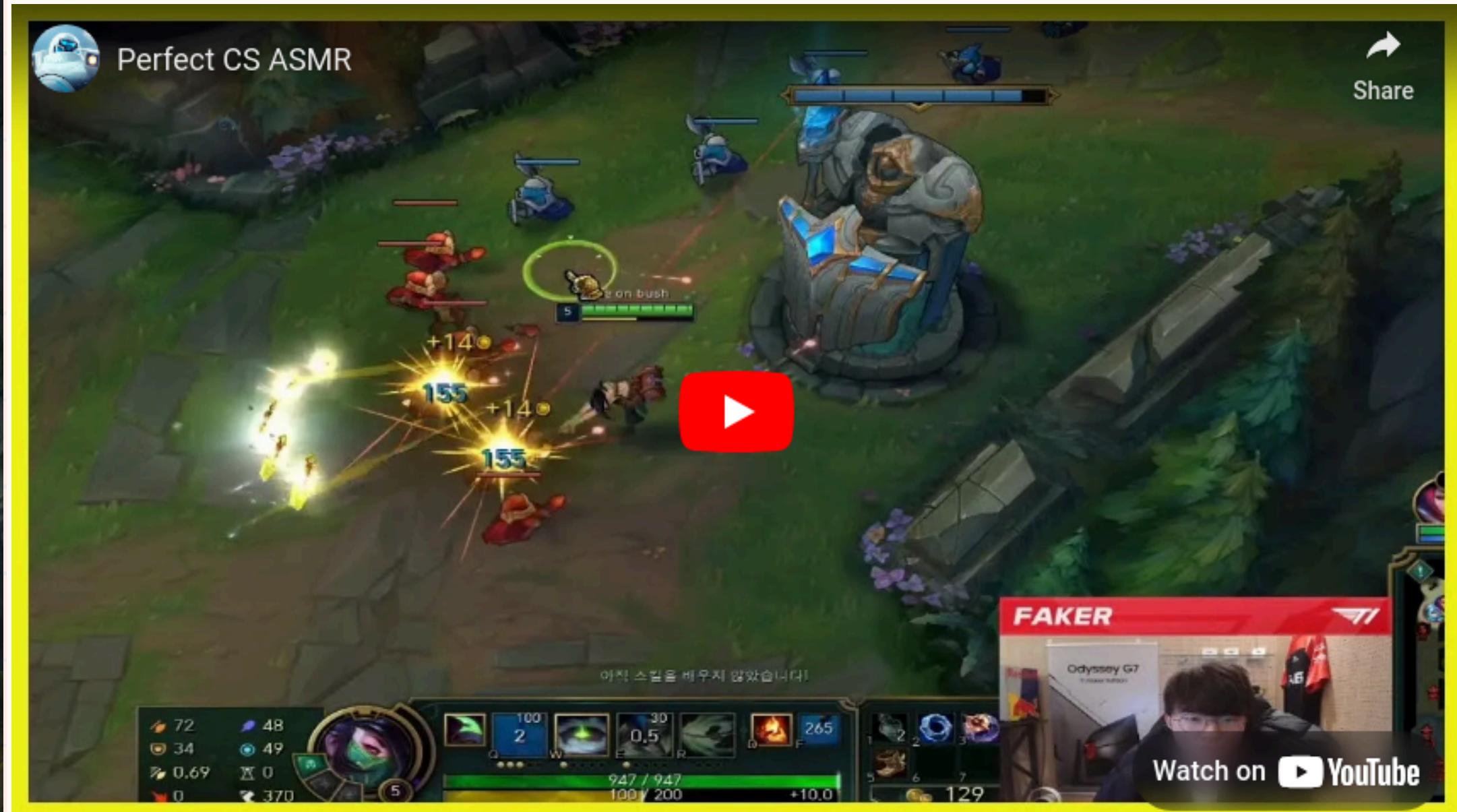
Faker's Sylas



- Là một trò chơi MOBA do Riot Games phát triển và ra mắt vào năm 2009.
- Nhiều giải đấu bán chuyên - chuyên nghiệp khu vực - thế giới
- Trò chơi đòi hỏi chiến thuật, kỹ năng cá nhân, và khả năng phối hợp nhóm.

Gameplay

- 2 team: xanh và đỏ
- Mỗi team có 5 người điều khiển 5 con tướng
- Mỗi vị tướng có bộ kỹ năng riêng biệt, phối hợp với đồng đội để đánh đổ nhà chính của đối phương.
- Người chơi có thể tiêu diệt lính, tướng, trụ, mục tiêu lớn để nhận vàng, kinh nghiệm, tăng cấp => Khai triển chiến thuật ở đây



Xếp hạng

Ranks & Queues

OVERVIEW

Climb the ranks in Solo/Duo and Flex queues to test your skill and earn rewards

TIERS

Iron > Bronze > Silver > Gold > Platinum

Emerald > Diamond > Master > Grandmaster > Challenger

DIVISIONS

IRON - DIAMOND

IV > III > II > I

NO DIVISIONS

Must be or higher to queue with

SOLO/DUO QUEUE

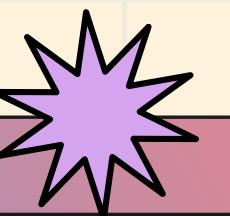
DUO RESTRICTIONS

	Iron	Bronze	Silver	Gold	Platinum	Emerald	Diamond	Master	Grandmaster	Challenger	IV-II	I	III-I	I
Iron														
Bronze														
Silver														
Gold														
Platinum														
Emerald														
Diamond														
Master														
Grandmaster														
Challenger														

FLEX QUEUE

SOLO ONLY

Các chỉ số quan trọng ảnh hưởng tới trận đấu



TỔNG QUAN THỬ THÁCH LỊCH SỬ ĐẤU HẠNG HIGHLIGHT

NGƯỜI CHƠI DẪN ĐẦU ĐƠN/ĐÔI
MÙA GIẢI 2024 - KỲ 3

CẬP NHẬT SAU: 23:20:44

Kết thúc sau 65 ngày 23 giờ 36 phút

CAO THỦ	ĐẠI CAO THỦ	THÁCH ĐẤU	
#	NGƯỜI CHƠI	T/B	ĐIỂM
1	千个伤心的理由 6	246/174	1730
2	mat hip ke t	187/116	1702
3	phuthuybom100	234/124	1501
4	千个伤心的理由 8	136/73	1438
5	千个伤心的理由 2	145/85	1335
6	emhyne	219/167	1107
7	my sweetie	122/73	1042
8	cyal8er29	114/62	978

Người chơi vẫn đang chiến đấu để đạt tới hạng này! Hãy xem lại sau!
Người chơi cần nằm trong top 300 và phải có ít nhất 500 ĐNG.

← → 🔍 ⌂ ⌂ ⌂

- Kills
- Deaths
- Assists
- KDA = (Kills + Assists)/Deaths
- Golds
- Minions

Cách lấy data

ChallengerSummonerID.txt

Lấy SummonerID

+ Code + Markdown

```
import time
import requests
api_key = 'RGAPI-415a160c-532f-41e2-9a53-78299b01cb10'

rank = 'CHALLENGER'
url = f"https://vn2.api.riotgames.com/lol/league-exp/v4/entries/RANKED_SOLO_5x5/{rank}/I?page=1&api_key={api_key}"
response = requests.get(url)
if response.status_code == 200:
    data = response.json()
    if data:
        for summoner in data:
            summonerID = summoner.get("summonerId")
            with open ("ChallengerSummonerID.txt", "a+") as file:
                file.write(summonerID + "\n")
    else:
        print("Failed to retrieve. Status Code:", response.status_code)
```

[5]

[14]

Python

riot_api.ipynb ChallengerSummonerID.txt

projectMAI > Challenger > ChallengerSummonerID.txt

```
1 wLyMvxPoLybfKA0C1aMrd51Zce03h0kObFmoBaXoyz12VCLcaR4MeMMefQ
2 X4zHZzBmvOPx34PX-5Wi3nybbCt1PQJG9onMRDUic83rjGtrotJNABXXMg
3 e80s49WbOZA7HyM013w9xVEPoLDfoR3VipVylizUaqFTpdaD_TQ67djG-w
4 8VZJ7Zk4yMJBkgb8rAylEYg0LydeQx5-alJQ5_8trCCea9xxxAwHKE3Gg
5 4Msyq8mRYxU2PXWrXYLqaFA9zVhv808B27629VFFu_dbvg4WxYFtJ6zy0Q
6 kGDxfsbJlywEKB51ZWxeyHJZgKwSvQ5RT9cERb5aTbggbJdvk7Za-f-Nw
7 Wi8J5BZfbpSMnSd0ACW8RQ80K1tJ-gshKhQEoSzuoSZGAG_CzlbaYC4iEQ
8 qxd0rPwF_Ybh7B05zp3H_qtN8GsMs34iVPv18bKEyOLxj8YWpmrb9Tqdbw
9 vMzx-Ifh4ms5zteJ1T9LXmRqiv71Yr96lwCMdz7b8G8FPqv6GPeXkwqh2g
10 ye3Hi532M1h7NZv0exUH_gNLfTpcaTnU71CGOVqyp8tyLaP0Lp0Ewpwwhw
11 DDRQBxat8fjdCijDADSnz8DLi0Eq3whY44dVwNwS9uFP6ThKR-Ffg_89uQ
12 cXKeCvdmHs6Nnb4BWDWes-6FmszhMqJYdfnkAnM9hj08mGhDnxDzcdm6xA
13 dPGp0wmQFeH0Ex-RMStCpCqEWS5mvVFNckhgx_j9dkmKtRMEW6fNO1v6_g
14 e-ic9UHKq-S-3V5alfHrMs7o2MARmVgC6q0_pT1URsjPNCTiULLm-0XdrQ
15 te-rk_xewvap9dbhuDZrsizs9U7eg5dCRR_77iHbXXFWf55jqDPkSXZF5Q
16 OQHD55qofWEHnBjYLh0FQNFxx6GVuAeeRKdP6vIZR_Gw5jhfz5ijk0IDZQ
17 IPYrvFH7fR0Xa--N0K8rx5RKGy0K_Xo4R-3mNiVEd7DcuhmTOoR-pBCxxA
18 9K5th1TDPyzsvIpXU8wJ2IFnadl9v4L99ruDdf_mHAyrEu3NTW-P-bZZEA
19 BBoGor1Mjk8KV9Kr5njchC_-BN9sCD3aPzcODED0Lwun_dW7JHqU-qITmQ
20 uLeD5q9obE5XpLmI7VqjDP4gcESgINC0Ggi4mRQhSIyBfmU4KQNSVK0lg
21 cRRoBtXx_VPJ_qgnEM-1EyXMrU9JcpmtJCgsiyJNAPrsmaScs21m5BrLw
22 Vtp1uCZrz41X-0z3yYJAjKNOf90fM1pEqFYLtYbqHF2HztLTLoCgherYg
23 mZJkMys3Ynv84yaC8JZytFfp3zQvXXVxnvUBGsT2e3DM-ZG5LsgIDch4uw
24 NJy1tPNyR4H2POAcftCbBH79ronvo9FHzrB2uCKqjJ64qDruAoLS5wIsg
25 qClxn8Q-rsSt7CwaFwTPoXj9Nz14M6r_r76RZ1bT_K20r3skcmJlkGsk8Q
26 1X5kamp5HbZrPfdK8mBd5XiXVElsbPqB-u0nbwBDUGHgxfpJOCMxfT40Sw
27 jqCcEXLy065MQZGjQ07IBH02YY9urmQ5e7P9XBzoT16MpM-zxaixVYNtyw
28 m3JxiqsGZOkBRitZPICTpuiyk7BASZ-flr0a-cSwGjDqcAcgaZSFk2fAw
29 Ki0mIChHsPmtG_zilrsMCJ8kGKoS5ucDuhQ2dku48x6XV0GYacUXEnMqza
30 TCXxphWprv-6clznbnE0zastNG_fn-tc40n1cYCaZ61sWZkZmXU56uwMpA
```

Cách lấy data

ChallengerPlayerPUUID.txt

```
Lấy PUUID

file_path = 'ChallengerSummonerID.txt'
with open(file_path, 'r') as file:
    for line in file:
        summoner_ID = line.strip()
        url1 = f"https://vn2.api.riotgames.com/lol/summoner/v4/summoners/{summoner_ID}?api_key={api_key}"
        response = requests.get(url1)
        if response.status_code == 200:
            data = response.json()
            player_puuid = data.get('puuid')
            with open('ChallengerPlayerPUUID.txt', 'a+') as file:
                file.write(player_puuid + "\n")
        else:
            print("Failed to retrieve. Status Code:", response.status_code)
```

```
riot_api.ipynb  ChallengerPlayerPUUID.txt

projectMAI > Challenger > ChallengerPlayerPUUID.txt
1  LUaGeYgfk9GEHLtELtR1A3HtdJo8ievYgD5ZiB_-trDEaEXnhgh6-jEYjLwFnJc-1Xc8LgPicCZm_A
2  xwivpF0h5BIJw3F7460KZMWQhSBTalunY0cjv8JfYQ44T1jHExyMalpcrb5s0dz1L50iRpI7YXbm0Q
3  LFWfQQ1TAFjMRih8dZM9RIEnwZ7i93z7sQjbHtTG_HMK9NZDWFzWLGY6Rq0cGwpAIZ1O-fZR6dnbuUQ
4  0561AjX28TETyMysqmv4KXQ8cUrXsVJqPCTPTtd5jeWzkqQ-OPMloyS0mkQG20zjis0-4ErU4nwQ
5  HpKIhcK4zVY4kyFVAjSZYG5NZ8HVmY3bPZabs5E_Aiyc3nqeVXUmuBBNt383DH1lZtkclyUVKm_SEA
6  0YqftXDEdbb-yE3buRersw7POyc8LIyIftdiJMibCZ_rtFq_7f7TLSLEKbsqx_jPOjt9uRP2Pd5eXg
7  ASwvi721fK6hTYdoySAs50R5cnWbNtPkZW36AhgHe5ue-RlmhqKGvb9B7iP_eOVtoZcgZqy2vVirBA
8  BR2wqq4JHeAM0FVINpZFaIC3ZjVK3z0LDgxkgWT38-0eBQ5MdT1xo2WjhWHo8LBgMNM0ZPyk9HaeCw
9  8gAwEnderaV_-ApYgR0av3J-X-KX11jH10IZYs1JP-6WfdM_yUFQzooYCTRJ6wDuseZFUiTvwD_RGQ
10 bckOkTWJA4y0rDcWpgQyJz1vXQFWCMDj5wCNZO3odsER16M_tM14ywUsrDUex8WNO15b6WSchQV1yQ
11 bVjgfqr3QX3MFSYFZ6kA9PteDG6APGC-UQRKjAkDZilMk_iReTfZRukw3XVGRWbTohesc4Q7QE5Ww
12 R7GoPiKO2P2Ln1OpbmMrU4u0JZy-cMv-bPLAtw-TStcyYYcfKQtqIMzm-YnHwI-GdrrNF0YHQw5nyg
13 A3fj-gRo9cPpkX8APUIxuvDe-Cjg1EahEGMz58BBY1DZ7ujT7QnRWnh5w8xdNq-4GLEfdpG-IHrEw
14 0axbvykvrhjT0vrTo_N07hM-PNynMfR9MHxE5ZRoxnYhXRWxXkbGLZoYnIZLXMF5X8I1M8HtfRk3oA
15 nlx2yMPR7hRLZNDK2tAUJr6R5_4NJtBrmDoXxhy25oFcIiDlmpTEldNr4CsdN4ay49f1GqgTkJQuFg
16 977DdhNjLIYFt20kjUQGd5Ft__CvT66GaGfbG3xKDyoltGv4rm0UU0cTmjshhZEpfDfzKu9LdGHGbA
17 mT7W0Qpw94i3PjUySu1G-iGK9EWvkFq9Plixh8eQE3IESzi2i8iR_1Nc0beIR8nnndh1uZCGB9r3MRw
18 C1v0WEmA0Eq-otK43SQ8Jc2C6YTCjcydSUQIs2UGVqYNgUEZ1Jxvethvnbmleq9baPfxZpeEiDjJqA
19 i76nKyGohE2ymsOLJnzCrBH1iH_fN3tyFZB8v0YL2s4dhm4lrXR-dSi7ogl_ld_tmwri6xcyb_HAg
20 XxpVrpyTW1RgvbJhUOA70wd1YS-EQ2XWD27TzCBhr7hS1TdN21Dg5tcDu_SX13Ti3GZy_FeL4gBYuw
21 wisLYM8ZC8Hief_hodQFB-dVNTbU53bY63bCP-ExtcxhJNjIKlvpAfFYknUDx13yhoMqbzT8HPNZA
22 7hogpdE1gwu2nFPWzdEcJHxInrmQd52LeJwg_FaydiueMxBcKL12ZiVME1HHRIVqF_B5CMRT4B3irQ
23 kHJgWx8jvquXPzoIZFXnG1iTTU1323EBXkvoQnGszss_dI5mDQqvWYaaBYUGzt1VIW-A15tuPNnhFw
24 2_BV2ohdn0iBqkfFSxXG1qUBkxPa17dHfb1E0kPStOXKJfUGNTLRLVhEUJxnXEGI_s0pTxXwpvnn8A
25 OIWd_4b0BXt7kAFtqR0Eiie89ieL7D5JFmC4cq8iFTz41ApAwHmSci34R1rAW98npIlH15QHgI3Xtg
26 rENO1RqxhHu_ZVWSH3IxKAOCn1LZ325xDX9A8gx_K_hACTgf6mnAf2LYoFsFUYpBxYYH_Qgpn7r5g
27 BTMRpzGwYRqvF3MITObmNGq6xnM5Nvlv9u8oLQQXBk2yFAAtRoF3ksQRuBLvWeUS7XxPOYRMB6o9Fw
28 10-atOEddUuWXd9ZYsK3_xbBdNBYqa6ChhAnK9zQtxAsHGuhUzv6B0t-OmspTJ-Ahr6LC2vxwYA1g
29 91qVlDenc9mam2w5J01H961qGuUmCBA97IA0KBGNpY2xrzHQ4C4oIj-6xqZpDgEZxRQqx8s71jkPHg
30 BbptBbGOvtKBEbtm3wp8I3kka5LEiwakUKA105T6m6wmEvk0axreIC8TxSV4ngIB1IM1cl5r_dxvSO
```

Cách lấy data

ChallengerPlayerMatchID.txt

Lấy MatchID

```
file_path1 = 'ChallengerPlayerPUUID.txt'
with open(file_path1, 'r') as file:
    for line in file:
        PUUID = line.strip()
        url = f"https://sea.api.riotgames.com/lol/match/v5/matches/by-puuid/{PUUID}/ids?queue=420&type=ranked&start=0&count=100"
        success = False
        while not success:
            response = requests.get(url)
            if response.status_code == 200:
                matchID = response.json()
                with open('ChallengerPlayerMatchID.txt', 'a+') as output_file:
                    output_file.write(str(matchID) + "\n")
                success = True
                time.sleep(6)
            else:
                print(f"Failed to retrieve data for PUUID. Status Code: {response.status_code}.")
                time.sleep(5)
```

[16]

Python

riot_api.ipynb ChallengerPlayerMatchID.txt

```
projectMAI > Challenger > ChallengerPlayerMatchID.txt
```

1 ['VN2_629899131', 'VN2_629870325', 'VN2_629849761', 'VN2_629832625', 'VN2_629819194', 'VN2_629797731', 'VN2_6296321354', 'VN2_6296073829', 'VN2_630054667', 'VN2_629365863', 'VN2_629306404', 'VN2_629207447', 'VN2_629168354', 'VN2_62915914', 'VN2_625419914', 'VN2_625412325', 'VN2_625400242', 'VN2_625378760', 'VN2_625349467', 'VN2_625321026', 'VN2_625318315', 'VN2_626304077', 'VN2_62628056', 'VN2_626266218', 'VN2_626241492', 'VN2_626206771', 'VN2_62624655454', 'VN2_626623130', 'VN2_626595020', 'VN2_626570901', 'VN2_626197172', 'VN2_626153621', 'VN2_62615093', 'VN2_62615276', 'VN2_628487854', 'VN2_628456929', 'VN2_627828595', 'VN2_627790009', 'VN2_627415093', 'VN2_6274143234', 'VN2_627129132', 'VN2_626958226', 'VN2_626773090', 'VN2_626333039', 'VN2_62630082126', 'VN2_630062957', 'VN2_630054667', 'VN2_630033734', 'VN2_630009586', 'VN2_629666032', 'VN2_629547191', 'VN2_629521906', 'VN2_629475884', 'VN2_628794510', 'VN2_628748771', 'VN2_628731323', 'VN2_629927590', 'VN2_629907044', 'VN2_629870325', 'VN2_629849761', 'VN2_629169348', 'VN2_629044024', 'VN2_625896453', 'VN2_625870770', 'VN2_625835172', 'VN2_625804086', 'VN2_625772490', 'VN2_625145975', 'VN2_630008248', 'VN2_629979849', 'VN2_629965170', 'VN2_629946380', 'VN2_629927062', 'VN2_629900079', 'VN2_629547191', 'VN2_629521906', 'VN2_629475884', 'VN2_629277733', 'VN2_629254179', 'VN2_629227735', 'VN2_629625784', 'VN2_629509440', 'VN2_629467557', 'VN2_629267181', 'VN2_629241582', 'VN2_629216909', 'VN2_630082126', 'VN2_630062723', 'VN2_629954662', 'VN2_629931358', 'VN2_629902050', 'VN2_629879215', 'VN2_628848624', 'VN2_628845349', 'VN2_628843204', 'VN2_628839121', 'VN2_628828622', 'VN2_628821534', 'VN2_627148452', 'VN2_627136960', 'VN2_627123355', 'VN2_627094779', 'VN2_624491551', 'VN2_624472516', 'VN2_629632280', 'VN2_629615095', 'VN2_629587516', 'VN2_629558557', 'VN2_629515869', 'VN2_629467557', 'VN2_629092679', 'VN2_630073829', 'VN2_630062723', 'VN2_630046294', 'VN2_629821729', 'VN2_629804769', 'VN2_629059208', 'VN2_630033734', 'VN2_630008248', 'VN2_629966201', 'VN2_629432587', 'VN2_629389463', 'VN2_627

Cách lấy data

```
[48]: import ast
import requests
import time

api_key = "RGAPI-415a160c-532f-41e2-9a53-78299b01cb10" # Thay thế bằng API key của bạn
cnt = 1

# Đọc tất cả các PUUID từ PlayerPUUID.txt
PlayerPUUID = 'ChallengerPlayerPUUID.txt'
PlayerMatchID = 'ChallengerPlayerMatchID.txt'
result = 'unprocessed_challenger.txt'
with open(PlayerPUUID, 'r') as puuid_file:
    target_puroids = [line.strip() for line in puuid_file]

# Đọc tất cả các mảng match IDs từ PlayerMatchID.txt
with open(PlayerMatchID, 'r') as match_file:
    match_id_arrays = [ast.literal_eval(line.strip()) for line in match_file]

requests_count = 0 # Đếm số lượng yêu cầu
start_time = time.time() # Thời gian bắt đầu

# Lặp qua mỗi target_puroid cùng với mảng match_id tương ứng
for target_puroid, match_ids in zip(target_puroids, match_id_arrays):
    for match_id in match_ids:
        # Kiểm tra và giới hạn số lượng yêu cầu mỗi giây và mỗi 2 phút
        current_time = time.time()

        # Nếu có 20 requests trong 1 giây, chờ đến khi đủ 1 giây
        if requests_count >= 20 and current_time - start_time < 1:
            time.sleep(1 - (current_time - start_time))
            start_time = time.time()
            requests_count = 0 # Reset đếm số yêu cầu

        # Nếu có 100 requests trong 2 phút, chờ thêm thời gian
        if requests_count >= 100 and current_time - start_time < 120:
            time.sleep(120 - (current_time - start_time))
            start_time = time.time()
            requests_count = 0 # Reset đếm số yêu cầu

url3 = f'https://sea.api.riotgames.com/lol/match/v5/matches/{match_id}?api_key={api_key}'

retry_attempts = 5 # Số lần thử lại khi gặp lỗi
while retry_attempts > 0:
    response3 = requests.get(url3)
    requests_count += 1 # Tăng đếm số yêu cầu

    if response3.status_code == 200:
        match_info = response3.json()
        participants = [p['puuid'] for p in match_info['info']['participants']]

        # Kiểm tra target_puroid trong danh sách participants
        if target_puroid in participants:
            index_of_target = participants.index(target_puroid)
            participant = match_info['info']['participants'][index_of_target]
            kills = participant['kills']
            deaths = participant['deaths']
            assists = participant['assists']
            kda = kills + assists if deaths == 0 else (kills + assists) / deaths
            total_gold = participant['goldEarned']
            total_minion = participant['totalMinionsKilled']

            # Ghi kết quả vào file
            with open(result, 'a+') as result_file:
                result_file.write(f"Kills: {kills} Deaths: {deaths} Assists: {assists} KDA: {kda:.2f} total_gold: {total_gold} total_minion: {total_minion}\n")

            print(f"Dã viết trận {cnt} cho {target_puroid} - ")
            cnt += 1
        else:
            print(f"PUUID {target_puroid} không tìm thấy trong trận đấu {match_id}")
            cnt += 1
    break # Nếu thành công, thoát khỏi vòng lặp retry
else:
    retry_attempts -= 1
    wait_time = 10 * (5 - retry_attempts) # Tăng thời gian chờ sau mỗi lần thất bại
    print(f"Failed to retrieve: {match_id}. Status Code: {response3.status_code}. Thù lại sau {wait_time} giây...")
    time.sleep(wait_time) # Chờ trước khi thử lại
```

ChallengerMatchID.txt

```
[VN2_629899131, 'VN2_629870325', 'VN2_629849761', 'VN2_629832625', 'VN2_629819194', 'VN2_629797731', 'VN2_629521906', 'VN2_629489564', 'VN2_629449360', 'VN2_629406514', 'VN2_629159866', 'VN2_628354513', 'VN2_628328873', 'VN2_628291500', 'VN2_627836846', 'VN2_627647834', 'VN2_627539578', 'VN2_626958226', 'VN2_626872422', 'VN2_626748215'] ['VN2_630073829', 'VN2_630054667', 'VN2_629365863', 'VN2_629306404', 'VN2_629207447', 'VN2_629168354', 'VN2_629107878', 'VN2_629069838', 'VN2_629047311', 'VN2_629019022', 'VN2_628821534', 'VN2_628817056', 'VN2_628803716', 'VN2_628798317', 'VN2_628773960', 'VN2_628748771', 'VN2_628722900', 'VN2_628180852', 'VN2_628159738', 'VN2_628140537]
```

ChallengerPlayerPUUID.txt

```
LUAGeYgfk9GEHLtELtR1A3HtdJo8ievYgD5ZiB-_trDEaEXnhgh6-jEYjLwFnJc-IXc8LgPicCZm_A  
xwivpF0h5BIJw3F7460KZM WQhSB TalunY0cjv8JfYQ44TljHExyMalpcrb5s0dz1L5  
0iRpI7YXbm0Q  
LWFfQQ1TAFjMRih8dZM9RIEnwZ7i93z7sQjbHtTG_HMK9NZDWFzWLGY6Rq  
0cGwpAIZ1O-fZR6dnbUQ
```

```
Dã viết trận 1 cho LUAgeYgfk9GEHLtELtR1A3HtdJo8ievYgD5ZiB-_trDEaEXnhgh6-jEYjLwFnJc-1Xc8LgPicCZm_A  
Dã viết trận 2 cho LUAgeYgfk9GEHLtELtR1A3HtdJo8ievYgD5ZiB-_trDEaEXnhgh6-jEYjLwFnJc-1Xc8LgPicCZm_A  
Dã viết trận 3 cho LUAgeYgfk9GEHLtELtR1A3HtdJo8ievYgD5ZiB-_trDEaEXnhgh6-jEYjLwFnJc-1Xc8LgPicCZm_A  
Dã viết trận 4 cho LUAgeYgfk9GEHLtELtR1A3HtdJo8ievYgD5ZiB-_trDEaEXnhgh6-jEYjLwFnJc-1Xc8LgPicCZm_A  
Dã viết trận 5 cho LUAgeYgfk9GEHLtELtR1A3HtdJo8ievYgD5ZiB-_trDEaEXnhgh6-jEYjLwFnJc-1Xc8LgPicCZm_A  
Dã viết trận 6 cho LUAgeYgfk9GEHLtELtR1A3HtdJo8ievYgD5ZiB-_trDEaEXnhgh6-jEYjLwFnJc-1Xc8LgPicCZm_A  
Dã viết trận 7 cho LUAgeYgfk9GEHLtELtR1A3HtdJo8ievYgD5ZiB-_trDEaEXnhgh6-jEYjLwFnJc-1Xc8LgPicCZm_A  
Dã viết trận 8 cho LUAgeYgfk9GEHLtELtR1A3HtdJo8ievYgD5ZiB-_trDEaEXnhgh6-jEYjLwFnJc-1Xc8LgPicCZm_A  
Dã viết trận 9 cho LUAgeYgfk9GEHLtELtR1A3HtdJo8ievYgD5ZiB-_trDEaEXnhgh6-jEYjLwFnJc-1Xc8LgPicCZm_A  
Dã viết trận 10 cho LUAgeYgfk9GEHLtELtR1A3HtdJo8ievYgD5ZiB-_trDEaEXnhgh6-jEYjLwFnJc-1Xc8LgPicCZm_A  
Dã viết trận 11 cho LUAgeYgfk9GEHLtELtR1A3HtdJo8ievYgD5ZiB-_trDEaEXnhgh6-jEYjLwFnJc-1Xc8LgPicCZm_A  
Dã viết trận 12 cho LUAgeYgfk9GEHLtELtR1A3HtdJo8ievYgD5ZiB-_trDEaEXnhgh6-jEYjLwFnJc-1Xc8LgPicCZm_A  
Dã viết trận 13 cho LUAgeYgfk9GEHLtELtR1A3HtdJo8ievYgD5ZiB-_trDEaEXnhgh6-jEYjLwFnJc-1Xc8LgPicCZm_A  
Dã viết trận 14 cho LUAgeYgfk9GEHLtELtR1A3HtdJo8ievYgD5ZiB-_trDEaEXnhgh6-jEYjLwFnJc-1Xc8LgPicCZm_A  
Dã viết trận 15 cho LUAgeYgfk9GEHLtELtR1A3HtdJo8ievYgD5ZiB-_trDEaEXnhgh6-jEYjLwFnJc-1Xc8LgPicCZm_A  
Dã viết trận 16 cho LUAgeYgfk9GEHLtELtR1A3HtdJo8ievYgD5ZiB-_trDEaEXnhgh6-jEYjLwFnJc-1Xc8LgPicCZm_A  
Dã viết trận 17 cho LUAgeYgfk9GEHLtELtR1A3HtdJo8ievYgD5ZiB-_trDEaEXnhgh6-jEYjLwFnJc-1Xc8LgPicCZm_A
```

```
Dã viết trận 1944 cho dvPtwcbyBxGo_gup_W_WaYsneJrr5lx416M2dGHVKybzFdSe2G9nPvaf6KLiygv80Yo9ZnnW6dXzw -  
Dã viết trận 1945 cho dvPtwcbyBxGo_gup_W_WaYsneJrr5lx416M2dGHVKybzFdSe2G9nPvaf6KLiygv80Yo9ZnnW6dXzw -  
Dã viết trận 1946 cho dvPtwcbyBxGo_gup_W_WaYsneJrr5lx416M2dGHVKybzFdSe2G9nPvaf6KLiygv80Yo9ZnnW6dXzw -  
Dã viết trận 1947 cho dvPtwcbyBxGo_gup_W_WaYsneJrr5lx416M2dGHVKybzFdSe2G9nPvaf6KLiygv80Yo9ZnnW6dXzw -  
Dã viết trận 1948 cho dvPtwcbyBxGo_gup_W_WaYsneJrr5lx416M2dGHVKybzFdSe2G9nPvaf6KLiygv80Yo9ZnnW6dXzw -  
Dã viết trận 1949 cho dvPtwcbyBxGo_gup_W_WaYsneJrr5lx416M2dGHVKybzFdSe2G9nPvaf6KLiygv80Yo9ZnnW6dXzw -  
Dã viết trận 1950 cho dvPtwcbyBxGo_gup_W_WaYsneJrr5lx416M2dGHVKybzFdSe2G9nPvaf6KLiygv80Yo9ZnnW6dXzw -  
Dã viết trận 1951 cho dvPtwcbyBxGo_gup_W_WaYsneJrr5lx416M2dGHVKybzFdSe2G9nPvaf6KLiygv80Yo9ZnnW6dXzw -  
Dã viết trận 1952 cho dvPtwcbyBxGo_gup_W_WaYsneJrr5lx416M2dGHVKybzFdSe2G9nPvaf6KLiygv80Yo9ZnnW6dXzw -  
Dã viết trận 1953 cho dvPtwcbyBxGo_gup_W_WaYsneJrr5lx416M2dGHVKybzFdSe2G9nPvaf6KLiygv80Yo9ZnnW6dXzw -  
Dã viết trận 1954 cho dvPtwcbyBxGo_gup_W_WaYsneJrr5lx416M2dGHVKybzFdSe2G9nPvaf6KLiygv80Yo9ZnnW6dXzw -  
Dã viết trận 1955 cho dvPtwcbyBxGo_gup_W_WaYsneJrr5lx416M2dGHVKybzFdSe2G9nPvaf6KLiygv80Yo9ZnnW6dXzw -  
Dã viết trận 1956 cho dvPtwcbyBxGo_gup_W_WaYsneJrr5lx416M2dGHVKybzFdSe2G9nPvaf6KLiygv80Yo9ZnnW6dXzw -  
Dã viết trận 1957 cho dvPtwcbyBxGo_gup_W_WaYsneJrr5lx416M2dGHVKybzFdSe2G9nPvaf6KLiygv80Yo9ZnnW6dXzw -  
Dã viết trận 1958 cho dvPtwcbyBxGo_gup_W_WaYsneJrr5lx416M2dGHVKybzFdSe2G9nPvaf6KLiygv80Yo9ZnnW6dXzw -  
Dã viết trận 1959 cho dvPtwcbyBxGo_gup_W_WaYsneJrr5lx416M2dGHVKybzFdSe2G9nPvaf6KLiygv80Yo9ZnnW6dXzw -  
Dã viết trận 1960 cho dvPtwcbyBxGo_gup_W_WaYsneJrr5lx416M2dGHVKybzFdSe2G9nPvaf6KLiygv80Yo9ZnnW6dXzw -  
Đã hoàn thành!
```

- Riot API limits:
- 20 requests every 1 second
- 100 requests every 2 minutes

Xử lý và lấy dữ liệu

unprocessed_challenger.txt

```
Kills: 10 Deaths: 2 Assists: 4 KDA: 7.00 total_gold: 10797 total_minion: 11
Kills: 13 Deaths: 5 Assists: 11 KDA: 4.80 total_gold: 15066 total_minion: 43
Kills: 12 Deaths: 2 Assists: 5 KDA: 8.50 total_gold: 10271 total_minion: 5
Kills: 5 Deaths: 1 Assists: 16 KDA: 21.00 total_gold: 9089 total_minion: 18
Kills: 5 Deaths: 5 Assists: 2 KDA: 1.40 total_gold: 6931 total_minion: 19
Kills: 3 Deaths: 4 Assists: 4 KDA: 1.75 total_gold: 5858 total_minion: 8
Kills: 5 Deaths: 5 Assists: 2 KDA: 1.40 total_gold: 6593 total_minion: 6
Kills: 8 Deaths: 6 Assists: 17 KDA: 4.17 total_gold: 13329 total_minion: 36
Kills: 8 Deaths: 7 Assists: 14 KDA: 3.14 total_gold: 13216 total_minion: 22
Kills: 10 Deaths: 5 Assists: 5 KDA: 3.00 total_gold: 12134 total_minion: 15
Kills: 6 Deaths: 9 Assists: 15 KDA: 2.33 total_gold: 13677 total_minion: 23
Kills: 3 Deaths: 1 Assists: 9 KDA: 12.00 total_gold: 8251 total_minion: 12
Kills: 9 Deaths: 0 Assists: 1 KDA: 10.00 total_gold: 8004 total_minion: 118
Kills: 7 Deaths: 7 Assists: 18 KDA: 3.57 total_gold: 16045 total_minion: 201
Kills: 9 Deaths: 4 Assists: 12 KDA: 5.25 total_gold: 15235 total_minion: 35
Kills: 1 Deaths: 8 Assists: 22 KDA: 2.88 total_gold: 9115 total_minion: 40
Kills: 6 Deaths: 4 Assists: 9 KDA: 3.75 total_gold: 11388 total_minion: 20
Kills: 4 Deaths: 6 Assists: 3 KDA: 1.17 total_gold: 8793 total_minion: 147
Kills: 3 Deaths: 2 Assists: 13 KDA: 8.00 total_gold: 11762 total_minion: 211
Kills: 11 Deaths: 1 Assists: 5 KDA: 16.00 total_gold: 11624 total_minion: 21
Kills: 1 Deaths: 3 Assists: 13 KDA: 4.67 total_gold: 5705 total_minion: 30
Kills: 1 Deaths: 0 Assists: 0 KDA: 1.00 total_gold: 2560 total_minion: 43
Kills: 8 Deaths: 4 Assists: 22 KDA: 7.50 total_gold: 15267 total_minion: 207
```

Dòng 1, Cột 1 | 148.351 ký tự

100%

Windows (CRLF)

UTF-8



```
# Đọc file PlayerMatchID.txt để xác định số phần tử trong mỗi mảng
match_counts = []
with open('PlayerMatchID.txt', 'r') as match_file:
    # Chuyển chuỗi dạng mảng thành một mảng thực sự
    array = ast.literal_eval(match_file.read())
    match_counts.append(len(array)) # Lưu số phần tử trong mỗi mảng

input_file = result
output_file = 'Challenger_final.txt'

with open(input_file, 'r') as file, open(output_file, 'a+') as output:
    # Ghi header
    output.write("Avg_Kills,Avg_Deaths,Avg_Assists,Avg_KDA,Avg_total_gold,Avg_total_minion\n")

    kills, deaths, assists, kda, total_gold, total_minion = 0, 0, 0, 0, 0, 0
    count = 0
    group_count = 0

    for line in file:
        # Trích xuất giá trị từ dòng
        parts = line.strip().split()
        kills += int(parts[1])
        deaths += int(parts[3])
        assists += int(parts[5])
        kda += float(parts[7])
        total_gold += int(parts[9])
        total_minion += int(parts[11])
        count += 1

        # Kiểm tra nếu đã đạt đủ số phần tử trong mảng tương ứng
        if count == match_counts[group_count]:
            avg_kills = kills / match_counts[group_count]
            avg_deaths = deaths / match_counts[group_count]
            avg_assists = assists / match_counts[group_count]
            avg_kda = kda / match_counts[group_count]
            avg_total_gold = total_gold / match_counts[group_count]
            avg_total_minion = total_minion / match_counts[group_count]

            # Ghi các giá trị trung bình vào file output
            output.write(f'{avg_kills:.2f},{avg_deaths:.2f},{avg_assists:.2f},{avg_kda:.2f},{avg_total_gold:.2f},{avg_total_minion:.2f}\n')

            # Reset biến đếm và tổng
            kills, deaths, assists, kda, total_gold, total_minion = 0, 0, 0, 0, 0, 0
            count = 0
            group_count += 1

print(f'Dã hoàn thành tính số liệu trung bình của {group_count} người chơi ({output_file.split('_')[0]}) rồi')
```

Đã hoàn thành tính số liệu trung bình của 98 người chơi Challenger rồi

challenger_final.txt

Tệp	Chỉnh sửa	Dạng xem
Avg_Kills,Avg_Deaths,Avg_Assists,Avg_KDA,Avg_total_gold,Avg_total_minion		
6.90,4.20,9.35,6.06,10858.90,50.55		
4.20,3.15,10.70,4.89,9178.30,121.70		
8.50,3.95,8.55,6.68,11876.90,32.30		
7.00,4.20,7.15,4.68,10614.00,25.90		
5.30,6.05,11.40,3.67,10012.70,27.75		
6.60,7.80,5.65,1.60,13146.30,162.35		
9.00,4.55,8.90,5.33,11525.75,18.45		
7.60,4.45,8.75,5.90,11869.15,25.90		
1.25,5.55,16.10,4.14,7636.45,32.75		
6.25,5.55,7.90,3.73,10772.35,142.45		
2.30,3.05,11.05,6.35,7383.55,52.85		
4.35,5.35,4.35,1.75,9450.15,163.90		
9.65,6.25,5.80,3.30,12809.95,177.15		
2.90,4.10,10.35,4.41,7366.35,25.55		
6.40,4.70,6.85,3.57,13147.45,210.20		
6.00,4.75,8.25,4.82,11819.40,196.95		
8.60,3.70,7.45,5.59,12689.15,40.45		
8.10,6.05,10.20,4.01,11284.15,22.65		
5.85,5.35,6.75,3.08,11197.05,179.25		
4.15,4.80,12.65,3.69,8888.70,34.85		
7.50,4.85,5.20,3.51,11795.75,193.10		
4.35,4.70,4.80,3.22,10069.60,177.40		

Dòng 1, Cột 1 | 3.549 ký tự

100%

Windows (CRLF)

UTF-8



Linear Regression là gì?

- Linear Regression là một thuật toán học có giám sát (supervised learning) trong Machine Learning (tệp data có gán output là Rank ứng với mỗi hàng)
- Là phương pháp thống kê dùng để ước lượng mối quan hệ giữa các biến độc lập (input features) và biến phụ thuộc (output target).
- Linear Regression giả định rằng sự tương quan giữa các biến là tuyến tính, từ đó tìm ra hàm tuyến tính tốt nhất để biểu diễn mối quan hệ này.
- Thuật toán này dự báo giá trị của biến output từ các giá trị của các biến đầu vào (Avg_Kills, ... Avg_total_minion)

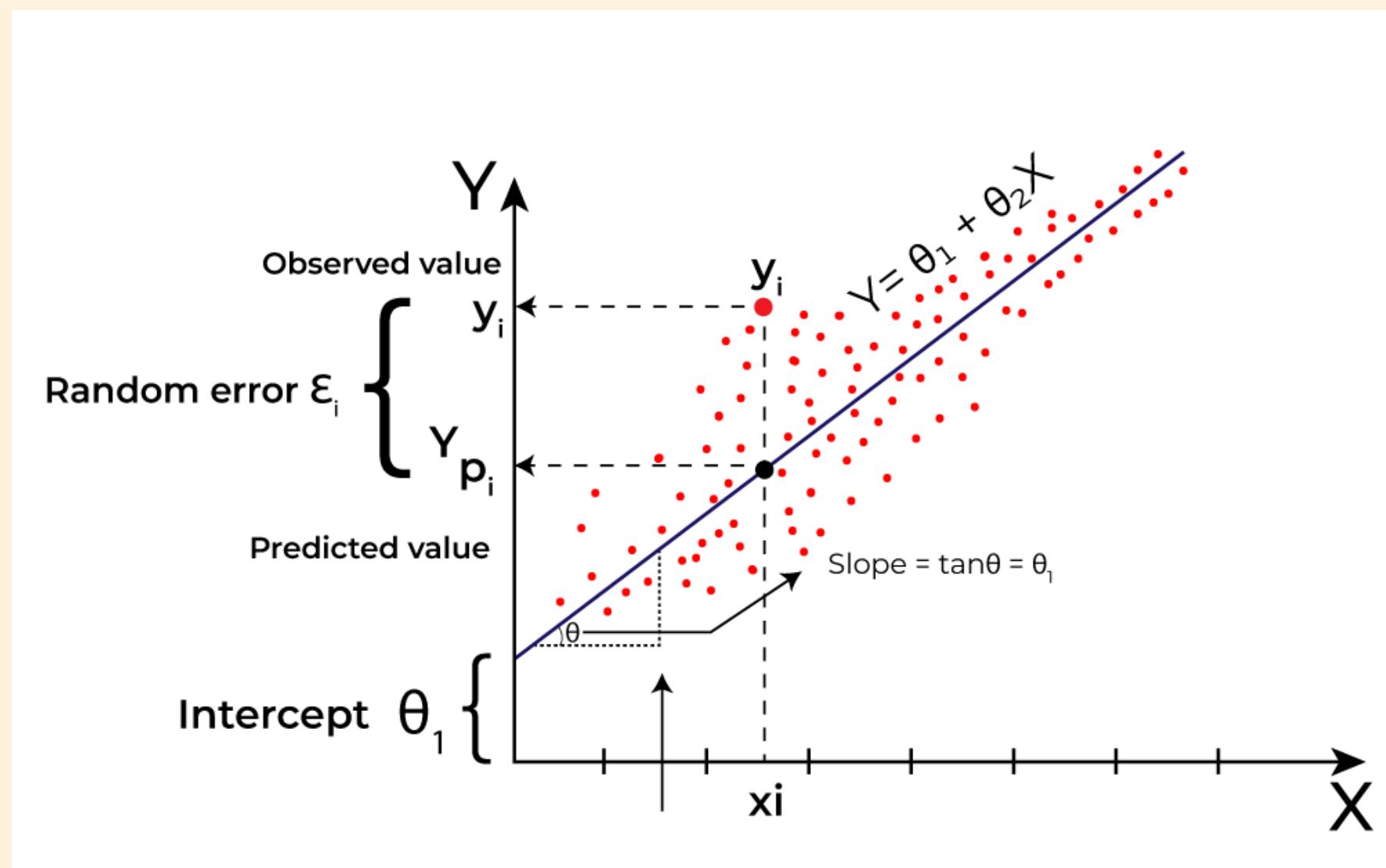
	Avg_Kills	Avg_Deaths	Avg_Assists	Avg_KDA	Avg_total_gold	Avg_total_minion	Rank
0	9.92	4.15	7.31	8.25	12732.31	114.54	0
1	4.15	4.85	6.50	3.20	9208.70	131.40	0
2	6.50	9.00	8.50	1.66	12305.95	164.05	0
3	6.70	5.95	6.55	3.58	11465.35	152.70	0
4	10.25	7.90	6.15	2.36	12844.50	155.60	0
...
866	7.60	6.85	8.35	2.55	11153.65	33.45	8
867	6.45	6.00	8.00	3.58	11494.40	172.30	8
868	5.90	5.65	5.90	2.70	10457.45	124.25	8
869	7.50	4.90	8.00	3.71	11815.15	182.20	8
870	5.00	5.90	7.30	3.16	9553.45	126.35	8

871 rows × 7 columns

Các loại Linear Regression

- Simple Linear Regression: Mô hình này chỉ có một biến độc lập (input feature) mô tả mối quan hệ tuyến tính giữa biến phụ thuộc (output target) và biến độc lập. Phương trình của Simple Linear Regression có dạng $y=a+bx+\epsilon$,
- Multiple Linear Regression: Mô hình này có nhiều hơn một biến độc lập, biểu diễn mối quan hệ tuyến tính giữa các biến độc lập và biến phụ thuộc. Phương trình của Multiple Linear Regression có dạng $y=a+b_1x_1+b_2x_2+\dots+b_nx_n+\epsilon$,

Với a là điểm giao với trục tung, b_1, b_2, \dots, b_n là các hệ số góc, và ϵ là sai số.



Công thức toán

- Chúng ta xem xét mô hình hồi quy tuyến tính đa biến, phương trình của nó có dạng:

$$y = a + b_1x_1 + b_2x_2 + \dots + b_nx_n + \epsilon$$

- Trong phương trình này, a là điểm giao với trục tung, b_1, b_2, \dots, b_n là các hệ số góc, x_1, x_2, \dots, x_n là các biến độc lập, và ϵ là sai số. Mục tiêu của chúng ta là tìm ra các hệ số của phương trình để tối thiểu hóa trung bình bình phương sai số (MSE):

$$\text{MSE} = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2$$

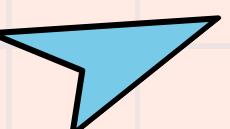
- Trong đó, m là số lượng mẫu dữ liệu, $y(i)$ là giá trị thực tế của biến phụ thuộc tại mẫu thứ i, và $\hat{y}(i)$ là giá trị dự đoán tại mẫu thứ i. Để tối thiểu hóa tổng bình phương sai số, chúng ta sử dụng phương pháp Gradient Descent.

Trích xuất, chuẩn hóa và chia tập dữ liệu

1. Input: dataframe đã gán nhãn

```
(  Avg_Kills  Avg_Deaths  Avg_Assists  Avg_KDA  Avg_total_gold  \
0      9.92      4.15       7.31      8.25    12732.31
1      4.15      4.85       6.50      3.20     9208.70
2      6.50      9.00       8.50      1.66    12305.95
3      6.70      5.95       6.55      3.58    11465.35
4     10.25      7.90       6.15      2.36    12844.50

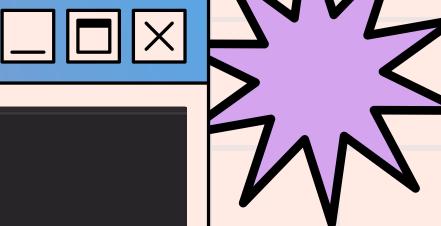
Avg_total_minion
0      114.54
1      131.40
2      164.05
3      152.70
4      155.60 ,
```



2. Gán data cho biến x, output y, xử lý chuẩn hóa

```
X=rank_data.iloc[:, :6]
y=rank_data.iloc[:, -1]
X=stats.zscore(X, axis=0)
y=stats.zscore(y, axis=0)
```

✓ 0.0s



3. Output: Dataframe đã được chuẩn hóa

```
(  Avg_Kills  Avg_Deaths  Avg_Assists  Avg_KDA  Avg_total_gold  \
0      1.773408   -1.167696   -0.467191   3.747877   1.066643
1     -0.953197   -0.640275   -0.787134  -0.359034  -1.175904
2      0.157292    2.486576    0.002848  -1.611439   0.795293
3      0.251802    0.188529   -0.767384  -0.049999   0.260306
4      1.929349    1.657772   -0.925381  -1.042164   1.138045

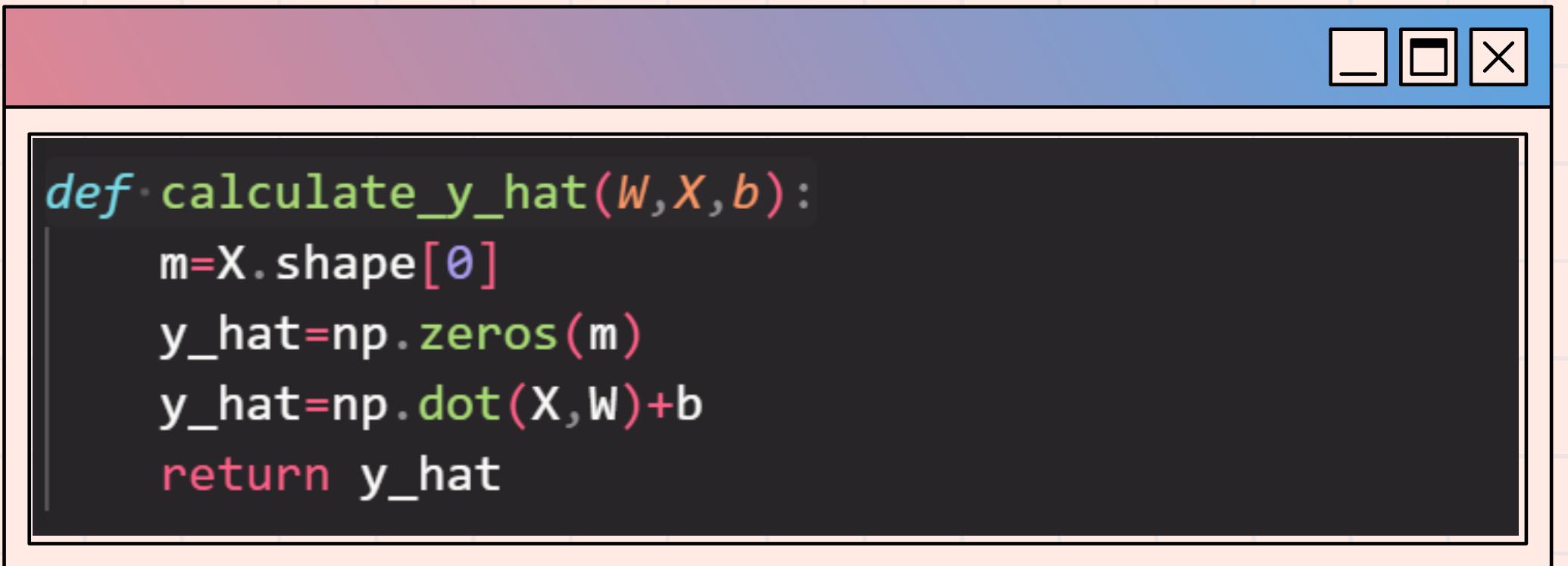
Avg_total_minion
0      -0.098048
1      0.207366
2      0.798811
3      0.593209
4      0.645742 ,
0     -1.562812
1     -1.562812
2     -1.562812
3     -1.562812
4     -1.562812
```

4. Tiến hành chia tập dữ liệu thành phần 80% train - 20% test

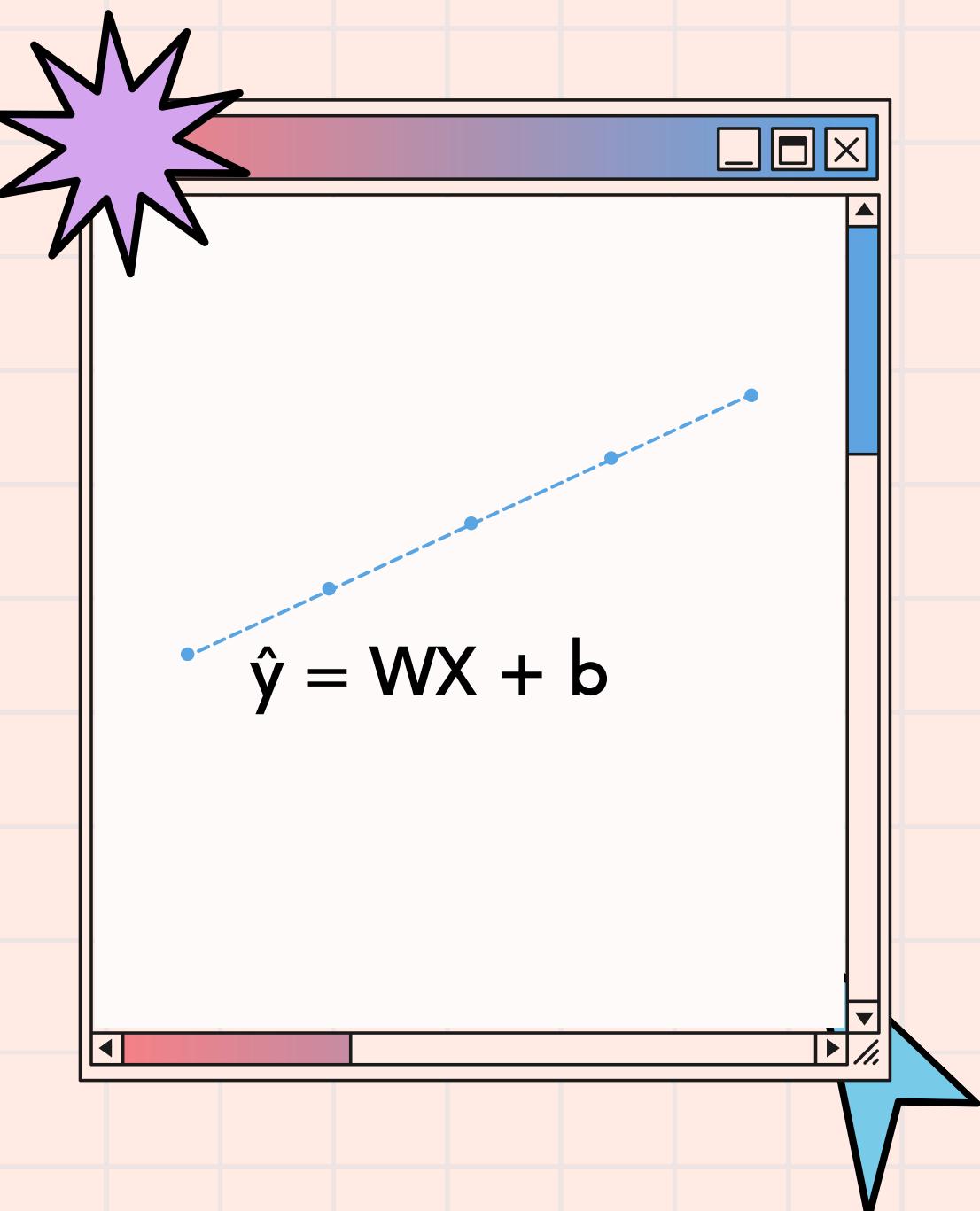
```
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test=train_test_split(X,y,random_state=42,test_size=0.2)

✓ 0.8s
```

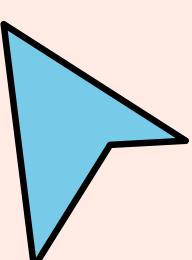
Hàm dự đoán đầu ra \hat{y}



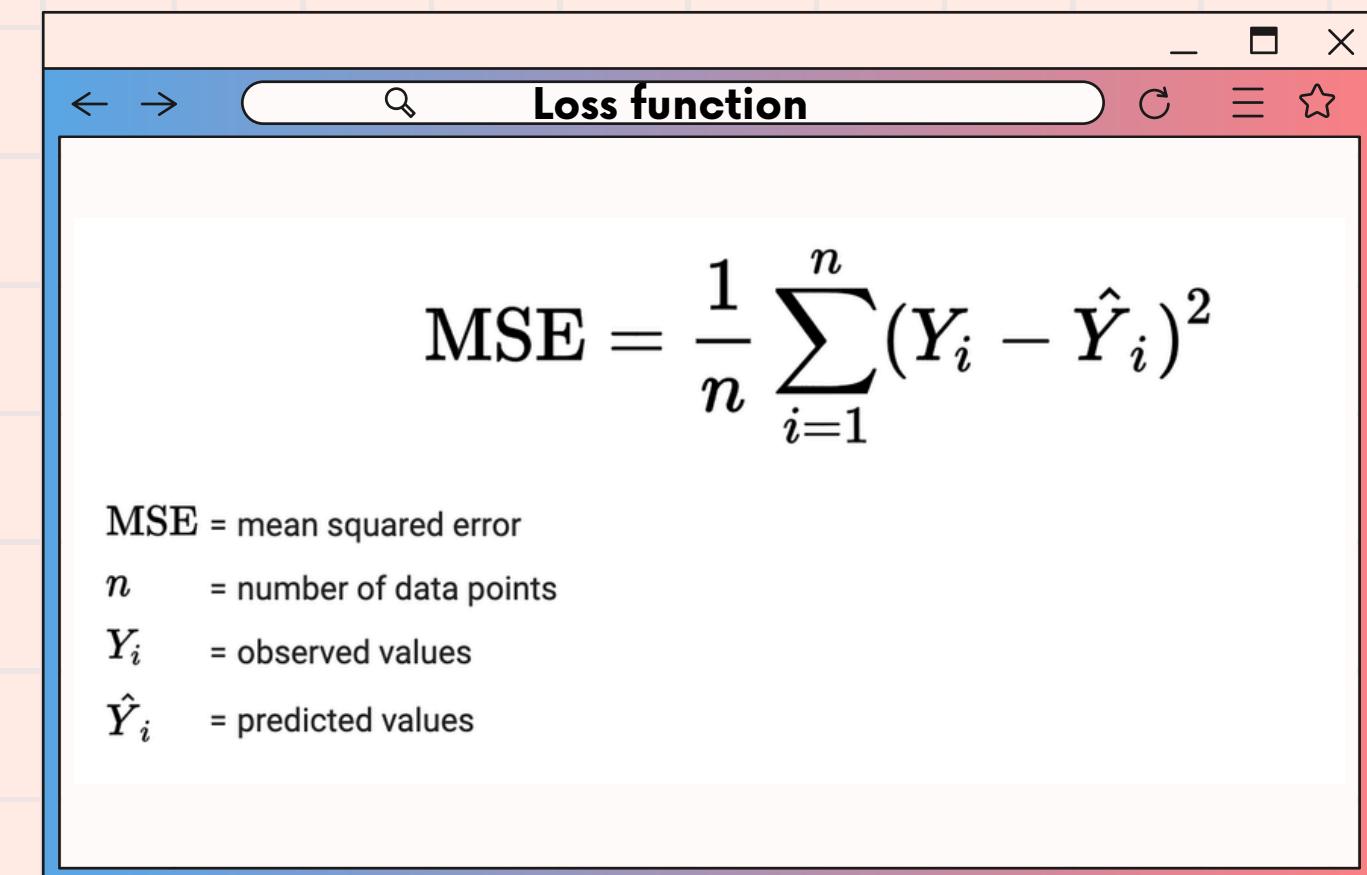
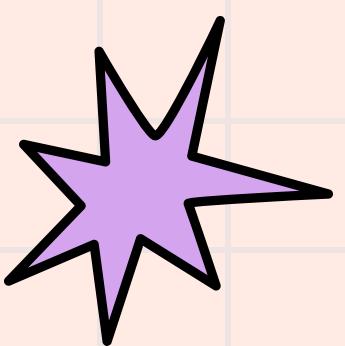
```
def calculate_y_hat(W,X,b):
    m=X.shape[0]
    y_hat=np.zeros(m)
    y_hat=np.dot(X,W)+b
    return y_hat
```



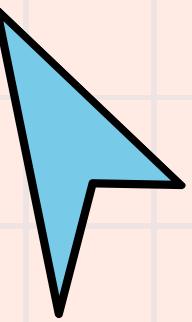
Hàm mất mát (Loss function)



```
def calculate_loss_function(y_hat,y):
    m=y_hat.shape[0]
    return np.sum((y - y_hat) ** 2) / (2 * m)
```



Tính gradient



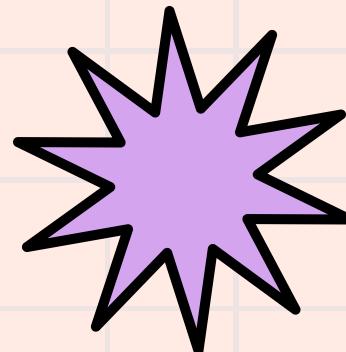
Untitled -TextEdit

File Edit View Help

```
def calculate_gradient_vector(W,X,b,Y):
    m=X.shape[0]
    n=W.shape[0]
    dW=np.zeros(n)
    db=0
    y_hat=calculate_y_hat(W,X,b)
    dW = np.dot(X.T, (Y - y_hat)) / m
    db = np.sum(Y - y_hat) / m
    return dW,db
```

Loss function gradient

$$\text{Loss} = \frac{1}{2m} \sum_{i=1}^m (y_i - \hat{y}_i)^2$$
$$\frac{\partial \text{Loss}}{\partial W} = -\frac{1}{m} \sum_{i=1}^m (y_i - \hat{y}_i) X_i$$
$$\frac{\partial \text{Loss}}{\partial b} = -\frac{1}{m} \sum_{i=1}^m (y_i - \hat{y}_i)$$



Cập nhật tham số bằng Gradient Descent và tính phương trình hồi quy tuyến tính

Untitled -TextEdit

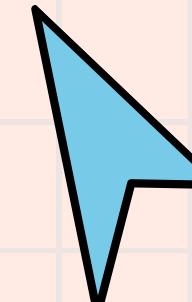
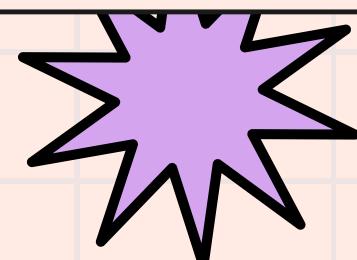
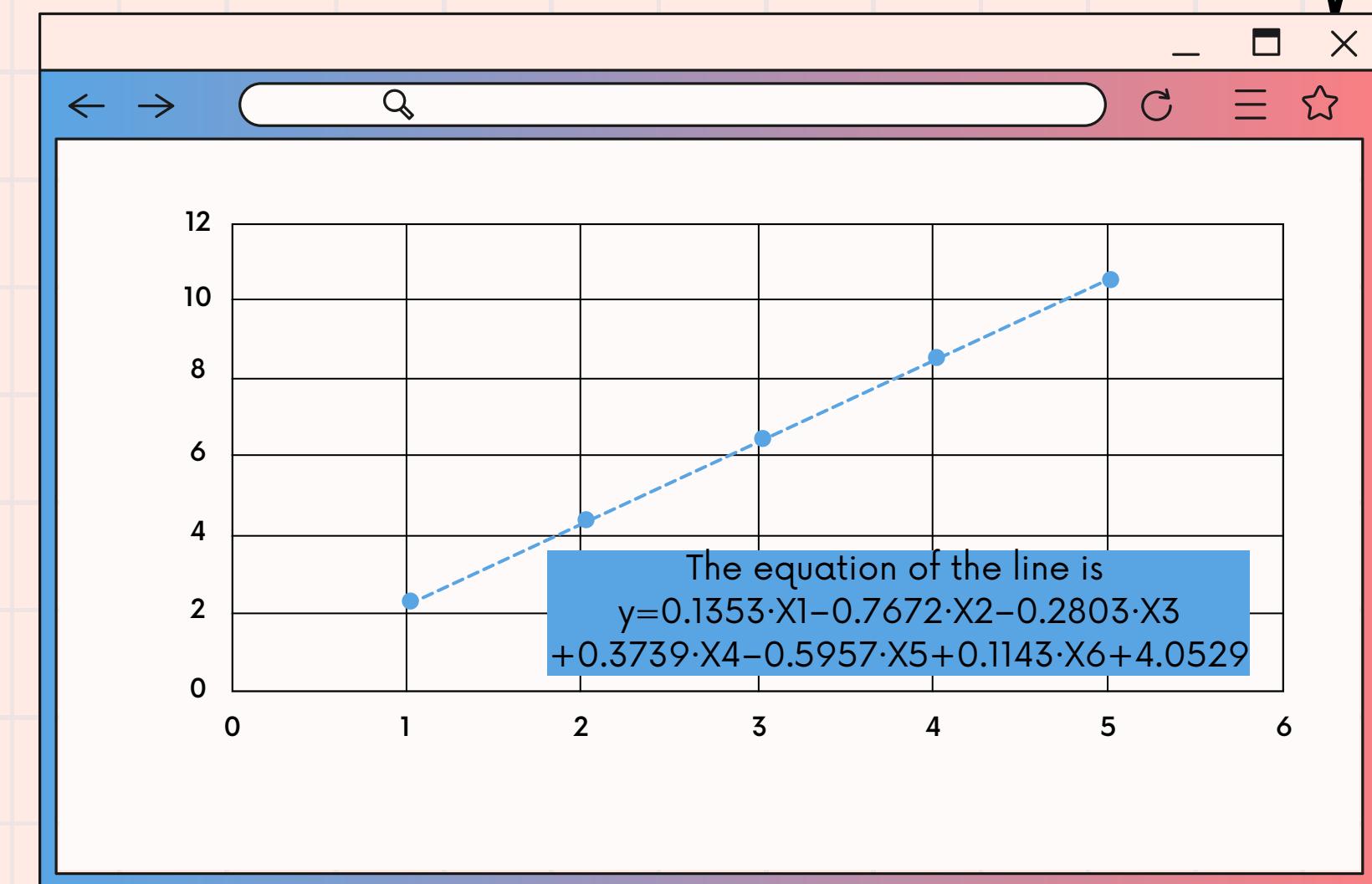
File Edit View Help

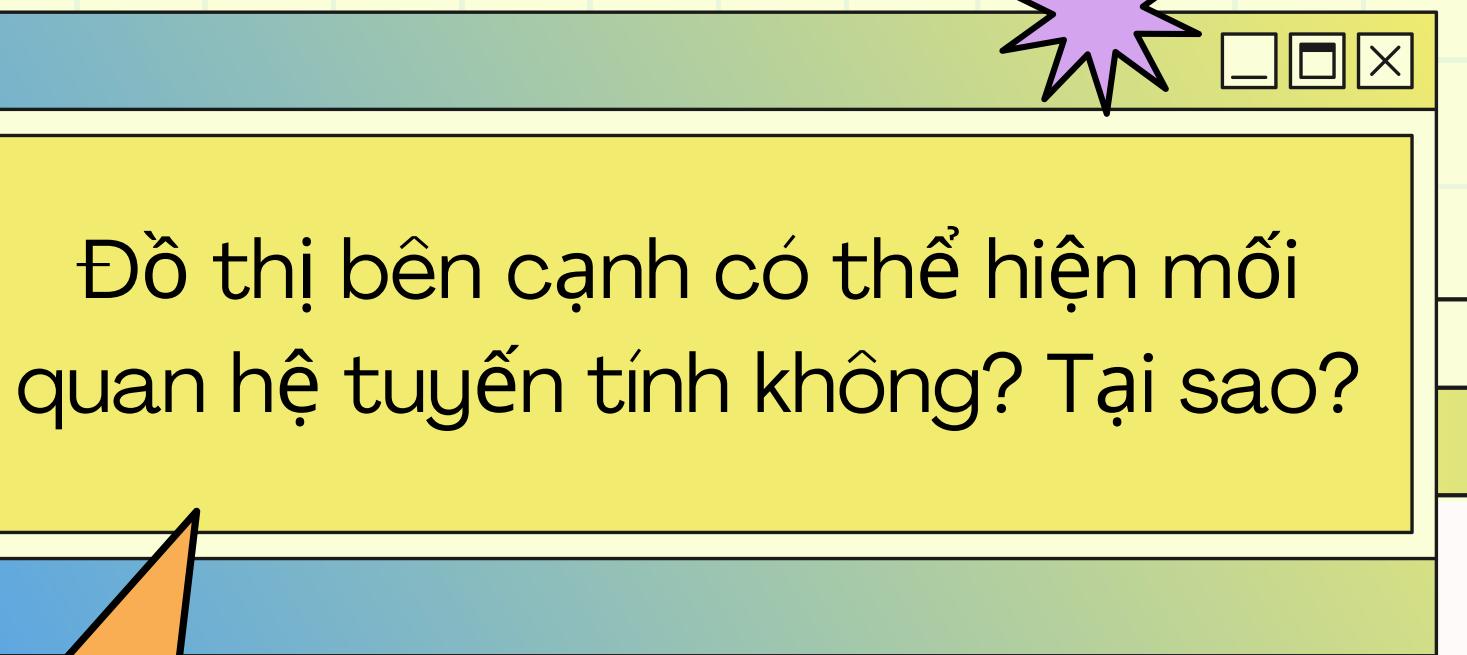
```
W=np.random.rand(*X.shape[1:])*0.01
b=np.random.rand()

✓ 0.0s

learning_rate=0.001
iterate_time=10000
✓ 0.0s

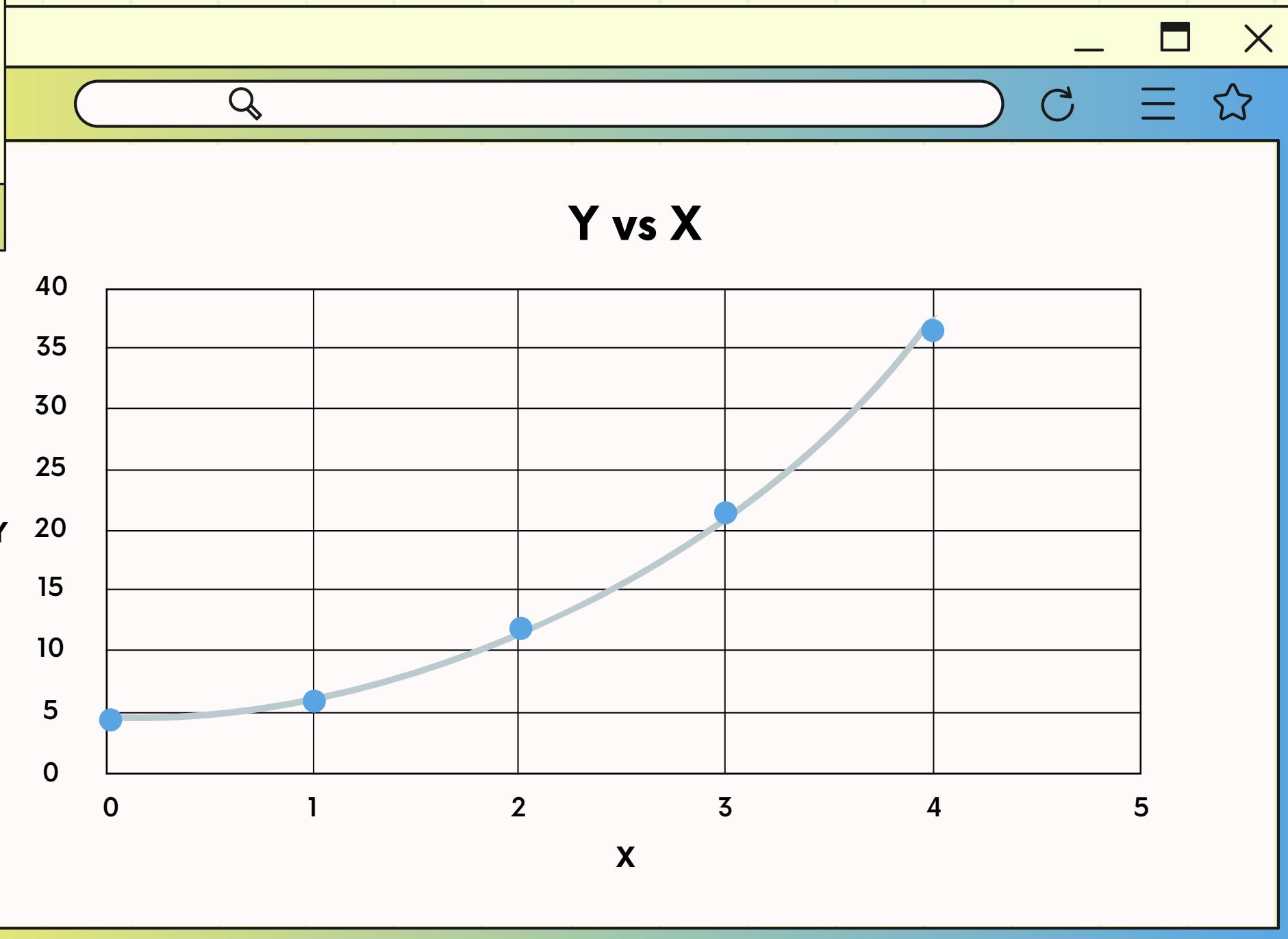
for k in range(iterate_time):
    dW,db=calculate_gradient_vector(W,X,b,y)
    W-=learning_rate*dW
    b-=learning_rate*db
    y_hat=calculate_y_hat(W,X,b)
    loss=calculate_loss_function(y_hat,y)
```





Không, đồ thị không cho thấy mối quan hệ tuyến tính vì đây không phải đường thẳng.

Đồ thị tạo thành một đường parabol.



Vì vậy ta sẽ có model regression của đa thức bậc cao hơn, trong đó sử dụng ý tưởng của công thức sau

$$\theta = (\Phi^T \phi)^{-1} \phi^T y$$

$$Y = \phi \theta$$

Đưa ý tưởng vào python

```
x_design_2 = [np.ones(x.shape[0])]  
for i in range(x.shape[1]):  
    x_design_2.append(x[:, i])  
for i in range(x.shape[1]):  
    for j in range(i, x.shape[1]):  
        x_design_2.append(x[:, i] * x[:, j])  
x_design_2 = np.column_stack(x_design_2)  
x_transpose_2 = x_design_2.T  
beta_2 = np.linalg.inv(x_transpose_2 @ x_design_2) @ x_transpose_2 @ y  
y_pred_2 = x_design_2 @ beta_2  
mse2 = np.mean((y - y_pred_2) ** 2)  
  
print("Mean Squared Error:", mse2)
```

Mean Squared Error: 0.6001606673747402

Cũng làm tương tự như thế với bậc 3

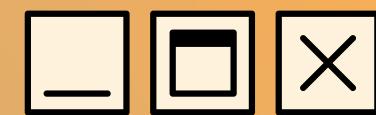
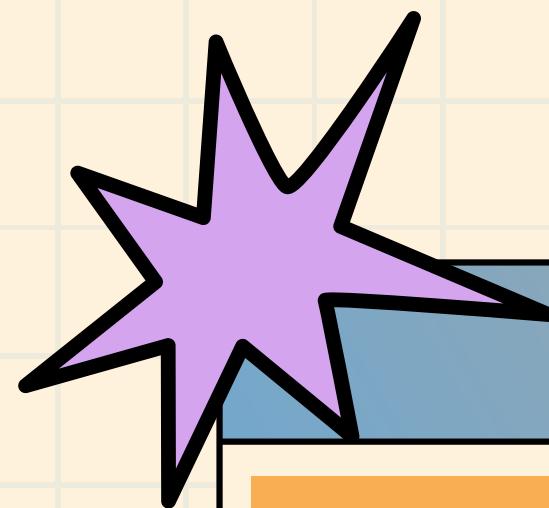
```
x_design_3 = [np.ones(x.shape[0])]  
lambda_reg_3=1  
for i in range(x.shape[1]):  
    x_design_3.append(x[:, i])  
for i in range(x.shape[1]):  
    for j in range(i, x.shape[1]):  
        x_design_3.append(x[:, i] * x[:, j])  
for i in range(x.shape[1]):  
    x_design_3.append(x[:, i] ** 3)  
    for j in range(i, x.shape[1]):  
        for k in range(j, x.shape[1]):  
            x_design_3.append(x[:, i] * x[:, j] * x[:, k])  
x_design_3 = np.column_stack(x_design_3)  
x_transpose_3 = x_design_3.T  
I_3 = np.identity(x_design_3.shape[1])  
beta_3 = np.linalg.inv(x_transpose_3 @ x_design_3 + lambda_reg_3 * I_3) @ x_transpose_3 @ y  
y_pred_3 = x_design_3 @ beta_3  
mse3 = np.mean((y - y_pred_3) ** 2)  
  
print("Mean Squared Error:", mse3)
```

Mean Squared Error: 0.539742559729076

Và bậc 4

```
x_design_4 = [np.ones(x.shape[0])]  
lambda_reg_4=1  
for i in range(x.shape[1]):  
    x_design_4.append(x[:, i])  
for i in range(x.shape[1]):  
    for j in range(i, x.shape[1]):  
        x_design_4.append(x[:, i] * x[:, j])  
for i in range(x.shape[1]):  
    x_design_4.append(x[:, i] ** 3)  
    for j in range(i, x.shape[1]):  
        for k in range(j, x.shape[1]):  
            x_design_4.append(x[:, i] * x[:, j] * x[:, k])  
for i in range(x.shape[1]):  
    x_design_4.append(x[:, i] ** 4)  
    for j in range(i, x.shape[1]):  
        for k in range(j, x.shape[1]):  
            for l in range(k, x.shape[1]):  
                x_design_4.append(x[:, i] * x[:, j] * x[:, k] * x[:, l])  
x_design_4 = np.column_stack(x_design_4)  
X_transpose_4 = X_design_4.T  
I_4 = np.identity(x_design_4.shape[1])  
beta_4 = np.linalg.inv(X_transpose_4 @ X_design_4 + lambda_reg_4 * I_4) @ X_transpose_4 @ y  
y_pred_4 = X_design_4 @ beta_4  
mse4 = np.mean((y - y_pred_4) ** 2)  
  
print("Mean Squared Error:", mse4)
```

- Ta có thể thấy rõ ràng rằng với các mô hình có bậc cao hơn, thì sai số càng giảm(MSE giảm từ 0.73 với bậc 1 xuống 0.44 với bậc 4), tức là mô hình sẽ ngày càng khớp nhiều hơn với data.
- Nhưng càng lên các bậc cao hơn, ta sẽ càng gặp nhiều vấn đề trong việc xử lí overfitting,cụ thể, với bậc 3 và 4 cần phải sử dụng hồi quy Ridge
- Đọc thêm về hồi quy Ridge và các phép biến đổi toán học tại : https://phamdinhkhanh.github.io/deepai-book/ch_ml/RidgedRegression.html



Thank you for listening!

