ĐẠI HỌC QUỐC GIA TP HCM

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

ĐỒ ÁN

Môn học: DỮ LIỆU LỚN

Đề tài: DỰ ĐOÁN BÀN THẮNG KỲ VỌNG TRONG MỖI TÌNH HUỐNG CỦA TRẬN ĐẤU BÓNG ĐÁ

Giảng viên hướng dẫn: **NGUYỄN HỒ DUY TRI**

Lớp: Dữ liệu lớn – IS405.N11.HTCL

Thành viên:

Lê Văn Long	-	19521783
Lê Ngọc Minh Thư	-	19522305
Lê Ngọc Tuấn	-	19522466
Nguyễn Trí Minh	-	19521847

TP Hồ Chí Minh tháng 12 năm 2022

MỤC LỤC

LỜI MỞ ĐẦU

Cùng với xu thế toàn cầu hóa và hội nhập quốc tế đang phát triển mạnh mẽ, ngày càng có nhiều dữ liệu được thu thập và lưu trữ dưới nhiều dạng khác nhau, gần hàng chục triệu tấm ảnh được lưu trữ mỗi ngày và hàng ngàn terabyte được sử dụng mỗi giây. Do đó, việc phân tích dữ liệu và đưa ra những dự đoán cần thiết là hết sức quan trọng và thiết yếu đối với xã hội ngày nay, từ đó giúp doanh nghiệp cũng như nhiều hoạt động quan trọng khác biết được hướng đi và nắm rõ sứ mệnh của mình.

Vì lý do đó nhóm em quyết định tìm hiểu về phân tích dữ liệu lớn và chọn kho dữ liệu về dự đoán các tình huống, sự kiện dẫn đến bàn thắng. Để bước đầu tìm hiểu về dữ liệu và phân tích nó dựa trên Spark và hệ thống tính toán song song.

LỜI CẢM ƠN

Trước khi đi vào nội dung phần báo cáo đồ án, đầu tiên xin cho phép nhóm của chúng em gửi lời cảm ơn sâu sắc đến thầy Nguyễn Hồ Duy Tri - giảng viên hướng dẫn môn học Dữ liệu lớn (IS405.N11.HTCL) đã cung cấp cho chúng em những kiến thức bổ ích và sự trợ giúp cần thiết trong suốt khoảng thời gian thực hiện đồ án, cũng như khoa Hệ thống Thông tin, Trường Đại học Công nghệ Thông tin, Đại học Quốc gia Thành phố Hồ Chí Minh đã tạo những điều kiện tốt nhất giúp cho chúng em có được cơ hội để thực hiện hoá đề tài và hoàn thiện đồ án này.

Cuối cùng, chúng em xin kính chúc Quý Thầy/Cô của khoa Hệ thống Thông tin cũng như thầy Nguyễn Hồ Duy Tri sức khoẻ dồi dào và thành công trên lĩnh vực của mình để sẵn sàng tiếp tục trên con đường truyền đạt kiến thức, truyền lửa và nhiệt huyết cho thế hệ mai sau. Xin trân trọng cảm ơn Quý Thầy/Cô rất nhiều.

NHẬN XÉT CỦA GIÁO VIÊN

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

PHÂN CHIA CÔNG VIỆC

STT	Họ và tên	MSSV	Nội dung công việc	Phần trăm đóng góp
1	Lê Văn Long	19521783	Đề xuất ý tưởng chính của đề tài, thực hiện chính phần lập trình	
2	Lê Ngọc Minh Thư	19522305	Hỗ trợ phần lập trình, soạn tài liệu, làm file trình chiếu	
3	Lê Ngọc Tuấn	19522466	Hỗ trợ phần lập trình, soạn tài liệu	
4	Nguyễn Trí Minh	19521847	Soạn tài liệu	

CHƯƠNG 1. GIỚI THIỆU TỔNG QUAN VỀ KHO DỮ LIỆU

1. PHÁT BIỂU BÀI TOÁN

Theo trang dữ liệu Wyscout, bàn thắng kỳ vọng (xG) là một mô hình machine learning dự đoán được sử dụng để đánh giá khả năng ghi bàn cho mỗi cú sút được thực hiện trong trận đấu.

Đối với các cú sút, mô hình xG tính toán xác suất ghi bàn dựa trên các thông số sự kiện:

- Vị trí của cú sút
- Vị trí của pha kiến tạo
- Chân hoặc đầu
- Loại hình kiến tạo
- Có một cú lừa bóng của một cầu thủ trên sân hoặc một thủ môn ngay lập tức trước khi thực hiện cú sút không?
- Có đến từ một tình huống cố định hay không?
- Cú sút đến từ một pha phản công hay nó xảy ra trong một quá trình chuyển đổi?
- Đánh giá của chuyên gia về mức độ nguy hiểm của cú sút

Xác suất nằm trong khoảng từ 0 đến 1. Một cú sút 0,1 xG có nghĩa là một cú sút như thế này có khả năng thành công khoảng 10%. Một cú sút 0,8 xG có nghĩa là một cú sút có tỷ lệ thành công khoảng 80%. Giá trị xG của 1 quả penalty được cố định là 0,76.

Trong đồ án này, nhóm em sử dụng dataset với các thông số như trên để đưa ra khả năng ghi bàn của một củ sút (hay còn gọi là Expected goal-XG- bàn thắng kỳ vọng).

2. PHÁT BIỂU VỀ DỮ LIỆU

2.1. Mô tả về dữ liệu

- Hơn 900.000 sự kiện từ 9.074 trận bóng đá trên khắp châu Âu, bao gồm Anh, Tây Ban Nha, Đức, Ý, Pháp từ năm 2011 đến năm 2017.
- Kho dữ liệu gồm 2 bảng, bảng thứ nhất là events.csv có 941009 dòng với 22 thuộc tính, bảng thứ hai là ginf.csv có 10112 với 18 thuộc tính.
- Link: https://www.kaggle.com/datasets/secareanualin/football-events

1.2. Thuộc tính của kho dữ liệu

- Kho dữ liệu events.csv

STT	Thuộc tính	Kiểu dữ liệu	Ý nghĩa của thuộc tính
1	id_odsp	Dạng chữ	Mã trận đấu (ví dụ: UFot0hit/)
2	id_event	Dạng chữ	Mã sự kiện (ví dụ: UFot0hit1)
3	sort_order	Dạng số	Thứ tự sự kiện trong một trận đấu (Từ 1 => 180)
4	time	Dạng số	Thời gian diễn ra trong một trận đấu (Từ 1 => 100)
5	text	Dạng chữ	Bình luận (ví dụ: Attempt missed. Mladen Petric,…)
6	event_type	Dạng số	Loại sự kiện (từ 0 => 11,99) 0: 'Announcement'- thông báo trong trận đấu, 1: 'Attempt' - nỗ lực ghi bàn, 2: 'Corner' - phạt góc, 3: 'Foul' - phạm lỗi, 4: 'Yellow card' - phạt thẻ vàng, 5: 'Second yellow card'- phạt thẻ vàng thứ 2, 6: 'Red card'- phạt thẻ đỏ, 7: 'Substitution' - thay người, 8: 'Free kick won' - được hưởng đá phạt,

			9: 'Offside' - việt vị, 10: 'Hand ball' - bóng chạm tay, 11: 'Penalty conceded' - được hưởng phạt đền, 99: 'NA'
7	event_type2	Dạng số	Loại sự kiện thứ hai (12 – 'Key Pass' - đường chuyền quyết định nhưng không thành bàn, 13 – 'Failed through ball' - đường chọc khe lỗi, 14-'Sending off' - (cầu thủ) bị thay ra, 15- 'Own goal'- (bàn thắng) phản lưới nhà)
8	side	Dạng số	Đội nhà và đội khách (1: 'Home'- đội nhà, 2: 'Away' - đội khách)
9	event_team	Dạng chữ	Đội tạo ra diễn biến
10	opponent	Dạng chữ	Đối thủ của đội tạo ra diễn biến
11	player	Dạng chữ	Tên cầu thủ tạo ra diễn biến
12	player2	Dạng chữ	Tên cầu thủ tạo ra diễn biến thứ 2
13	player_in	Dạng chữ	Cầu thủ được thay vào
14	player_out	Dạng chữ	Cầu thủ được thay ra
15	shot_place	Dạng chữ	Nơi sút bóng (1:'Bit too high'- hơi cao, 2:'Blocked'- bị chặn , 3:'Bottom left corner'- góc dưới bên trái, 4:'Bottom right corner'- góc dưới bên phải, 5:'Centre of the goal' - giữa khung thành, 6:'High and wide' - lên cao trên khung thành hoặc xa khung thành, 7:'Hits the bar' - trúng cột/xà ngang, 8:'Misses to the left ' - (sút) trượt ở bên trái, 9:'Misses to the right' - (sút) trượt ở bên phải, 10:'Too high'- quá cao, 11:'Top centre of the goal' - giữa bên trên khung thành, 12:'Top left corner'- góc trên bên trái, 13:'Top right corner'- góc trên bên phải,

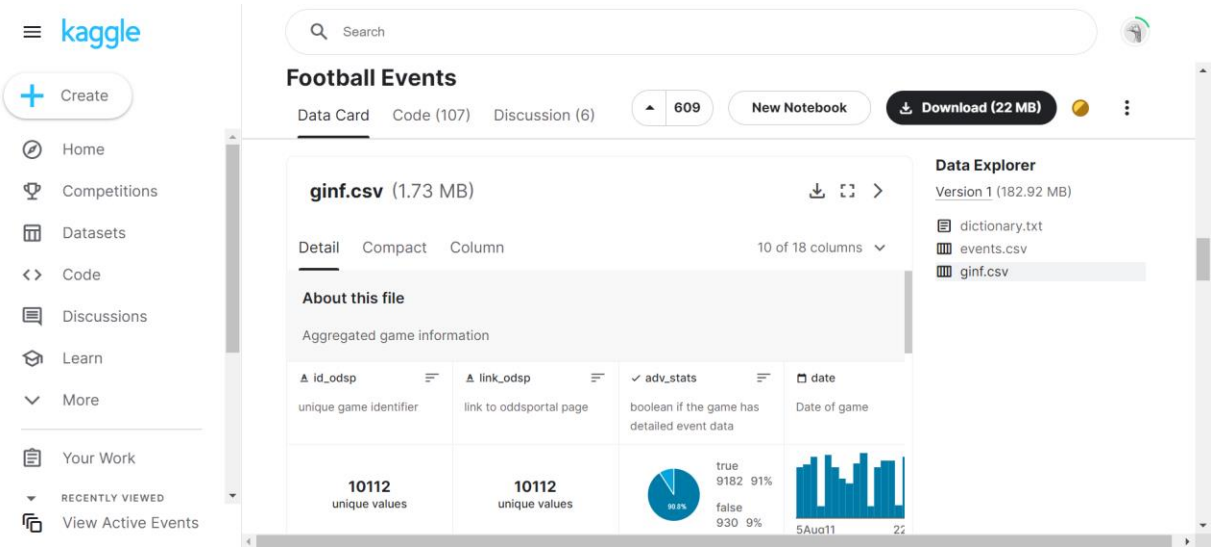
			0:'NA')
16	shot_outcome	Dạng số	4 loại kết quả (1:'On target' - trúng đích, 2:'Off target' - không trúng đích, 3:'Blocked' - bị chặn, 4:'Hit the bar' - trúng cột/xà ngang, 0:'NA'-không có dữ liệu)
17	is_goal	Dạng số	Dạng số (1-sút trúng, 0-sút không trúng)
18	location	Dạng chữ	Vị trí xảy ra trên sân (1:Attacking half' - nửa sân của đội đang tấn công, 2:'Defensive half' - nửa sân của đội đang phòng thủ, 3:'Centre of the box' - giữa vòng cấm, 4:'Left wing'- cánh trái, 5:'Right wing'- cánh phải, 6:'Difficult angle and long range' - góc sút khó và ở khoảng cách xa, 7:'Difficult angle on the left' - góc sút khó bên trái, 8:'Difficult angle on the right' - góc sút khó bên phải, 9:'Left side of the box' - bên trái vòng cấm, 10:'Left side of the six yard box' - bên trái khu vực chữ nhật trước khung thành, 11:'Right side of the box' - bên phải vòng cấm, 12:'Right side of the six yard box' - bên phải khu vực chữ nhật trước khung thành, 13:'Very close range'- khoảng cách rất gần, 14:'Penalty spot' - chấm phạt đền, 15:'Outside the box' - ngoài vòng cấm, 16:'Long range'- khoảng cách xa, 17:'More than 35 yards'- hơn 35 dặm, 18:'More than 40 yards'- hơn 40 dặm, 19:'Not recorded'- không được ghi lại, 0:'NA')

19	bodypart	Dạng chữ	Phần cơ thể chạm bóng (1- Right foot-Chân phải, 2- Left foot -Chân trái, 3-Head-Đầu)
20	assist_method	Dạng số	5 phương thức kiến tạo (từ 0=>4) 0:'None' 1:'Pass' - Chuyền trực tiếp, 2:'Cross' - Tạt bóng, 3:'Headed pass' - Chuyền bằng đầu, 4:'Through ball' - Chọc khe
21	situation	Dạng chữ	4 loại tình huống (1:'Open play' - Bóng sống, 2:'Set piece' - Bóng trở lại trạng thái open play sau khi trận đấu dừng lại, 3:'Corner - Phạt góc', 4:'Free kick' - Phạt trực tiếp)
22	fast_break	Dạng số	Trận đấu có bị dừng hay không

- Kho dữ liệu ginf.csv

STT	Thuộc tính	Kiểu dữ liệu	Ý nghĩa của thuộc tính
1	id_odsp	Dạng chữ	Mã trận đấu (ví dụ: UFot0hit/)
2	link_odsp	Dạng chữ	Liên kết đến trang Oddportal
3	adv_stats	Dạng boolean	True: trò chơi có sự kiện chi tiết, False: trò chơi không có sự kiện chi tiết
4	date	Dạng ngày tháng	Ngày bắt đầu đá
5	league	Dạng chữ	Câu lạc bộ giải đấu
6	season	Dạng số	Năm chơi
7	country	Dạng chữ	Nước chủ nhà của giải đấu ('germany', 'france', 'england', 'spain', 'italy')

8	ht	Dạng chữ	Đội chủ nhà
9	at	Dạng chữ	Đội khách
10	fthg	Dạng số	Bàn thắng trên sân nhà cả trận
11	ftag	Dạng số	Bàn thắng trên sân khách cả trận
12	odd_h	Dạng số	Tỷ lệ cược thị trường thắng nhà cao nhất
13	odd_d	Dạng số	tỷ lệ cược thị trường hòa cao nhất
14	odd_a	Dạng số	Tỷ lệ cược sân khách cao nhất
15	odd_over	Dạng chữ	Cao nhất trên 2,5 tỷ lệ cược thị trường
16	odd_under	Dạng chữ	Tỷ lệ cược thị trường dưới 2,5 cao nhất
17	odd_bts	Dạng chữ	Điểm cao nhất cả hai đội để ghi tỷ lệ cược thị trường
18	odd_bts_n	Dạng chữ	Điểm cao nhất cả hai đội KHÔNG ghi bàn được



1.3. Kho dữ liệu đã xử lý

- Dữ liệu sau khi xử lý và gom bảng ta được 941009 dòng và 33 cột thuộc tính để sử dụng cho việc phân tích.

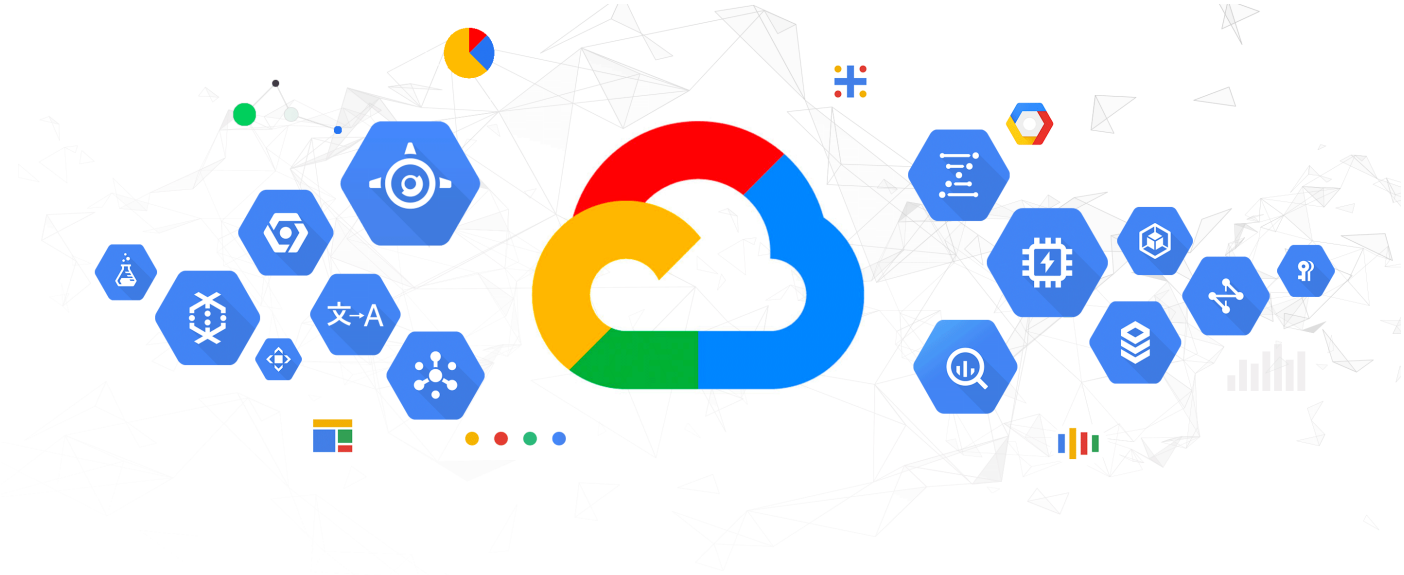
STT	Thuộc tính	Kiểu dữ liệu	Ý nghĩa của thuộc tính

1	game_id	Dạng số	Mã trận đấu
2	id_event	Dạng chữ	Mã sự kiện
3	sort_order	Dạng số	Thứ tự sự kiện trong một trận đấu (Từ 1 => 180)
4	time	Dạng số	Thời gian diễn ra trong một trận đấu (Từ 1 => 100)
5	event_type	Dạng số	Loại sự kiện (từ 1 => 11)
6	event_type_str	Dạng chữ	Cột chuyển đổi từ event_type
7	event_type2	Dạng số	Loại sự kiện thứ hai (12 - Key Pass, 13 - Failed through ball, 14-Sending off, 15-Own goal)
8	event_type2_str	Dạng chữ	Cột chuyển đổi từ event_type2
9	side	Dạng số	Đội nhà và đội khách (1: đội nhà, 2: đội khách)
10	side_str	Dạng chữ	Cột chuyển đổi từ side
11	event_team	Dạng chữ	Đội tạo ra diễn biến
12	opponent	Dạng chữ	Đối thủ của đội tạo ra diễn biến
13	player	Dạng chữ	Tên cầu thủ tạo ra diễn biến
14	player2	Dạng chữ	Tên cầu thủ tạo ra diễn biến thứ 2
15	player_in	Dạng chữ	Cầu thủ được thay vào
16	player_out	Dạng chữ	Cầu thủ được thay ra
17	shot_place	Dạng số	Nơi sút bóng
18	shot_place_str	Dạng chữ	Chuyển đổi từ shot_place
19	shot_outcome	Dạng số	4 loại kết quả (1-trúng đính, 2-không trúng đích, 3-bị chặn, 4-đụng vào khung thành)
20	shot_outcome_str	Dạng chữ	Chuyển đổi từ shot_outcome
21	is_goal	Dạng số	Dạng số (1-sút trúng, 0-sút không trúng)

22	location	Dạng số	Vị trí xảy ra trên sân
23	location_str	Dạng chữ	Chuyển đổi từ location
24	bodypart	Dạng số	Phần cơ thể chạm bóng (1-Chân phải, 2-Chân trái, 3-Đầu)
25	bodypart_str	Dạng chữ	Chuyển đổi từ bodypart
26	assist_method	Dạng số	5 phương thức kiến tạo (từ 0=>4)
27	assist_method_str	Dạng chữ	Chuyển đổi từ assist_method
28	situation	Dạng số	4 loại (1-Bóng sống, 2-Đá phạt, 3-Đá góc, 4-Đá phát trực tiếp)
29	situation_str	Dạng chữ	Chuyển đổi từ situation
30	country_code	Dạng chữ	Chuyển đổi từ country
31	country_ID	Dạng chữ	Chuyển đổi từ country_code
32	date	Dạng ngày tháng	Ngày bắt đầu đá
33	time_bin	Dạng số	Gom từ time

CHƯƠNG 2. CÀI ĐẶT PHẦN MỀM

1. GIỚI THIỆU GOOGLE CLOUD PLATFORM



Google Cloud Platform viết tắt là (GCP) là nền tảng điện toán đám mây do Google cung cấp, nền tảng này bao gồm một loạt các dịch vụ được lưu trữ để tính toán, lưu trữ và phát triển ứng dụng chạy trên phần cứng của Google. Các dịch vụ Google Cloud có thể được truy cập bởi các nhà phát triển phần mềm, quản trị viên hoặc các chuyên gia CNTT qua internet hoặc thông qua kết nối mạng.

Google Cloud cung cấp các dịch vụ về máy tính, mạng, lưu trữ, big data, machine learning và IoT, cũng như các công cụ khác dành cho nhà phát triển bảo mật và quản lý Cloud. Một số ứng dụng của Google Cloud bao gồm:

- **Google Compute Engine**:Là cơ sở hạ tầng dưới dạng dịch vụ(laaS) cung cấp cho người dùng các phiên bản VM để lưu trữ khối lượng công việc.
- **Google Cloud Storage**:Là một nền tảng lưu trữ đám mây được thiết kế để lưu trữ các tập dữ liệu lớn, không có cấu trúc.

- **Google Kubernetes Engine (GKE)**: là hệ thống quản lý và điều phối cho vùng chứa Docker và các cụm vùng chứa chạy trong các dịch vụ public cloud services của Google.
- **Bộ hoạt động của Google Cloud**: Trước đây là Stackdriver là một bộ công cụ tích hợp để theo dõi, ghi lịch sử và báo cáo về các dịch vụ được quản lý thúc đẩy các ứng dụng và hệ thống trên Google Cloud.
- **Điện toán không máy chủ**: Cung cấp các công cụ và dịch vụ để thực thi khối lượng công việc dựa trên sự kiện.
- **Cơ sở dữ liệu**: Là một bộ sản phẩm cơ sở dữ liệu được cung cấp dưới dạng dịch vụ được quản lý hoàn toàn, bao gồm Cloud Bigtable cho khối lượng công việc quy mô lớn, độ trễ thấp.
- Ở đồ án, này nhóm sử dụng chức năng **Google Compute Engine,** phiên bản trial để hỗ trợ tạo các máy ảo Ubuntu.

2. CÀI ĐẶT TRÊN GOOGLE CLOUD PLATFORM

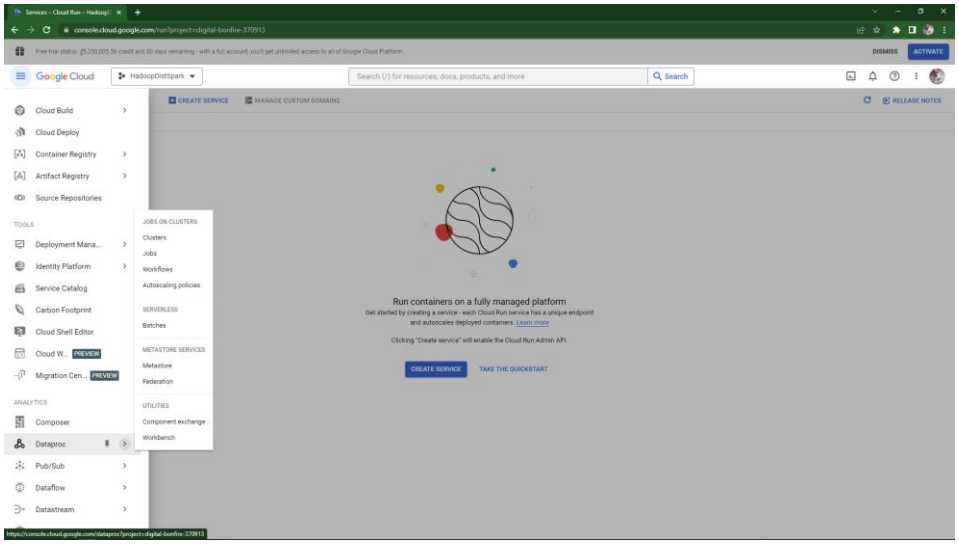
2.1 Cách thức đăng ký tài khoản Google VPS

- **Bước 1**: Đăng nhập gmail và vào đường dẫn https://cloud.google.com

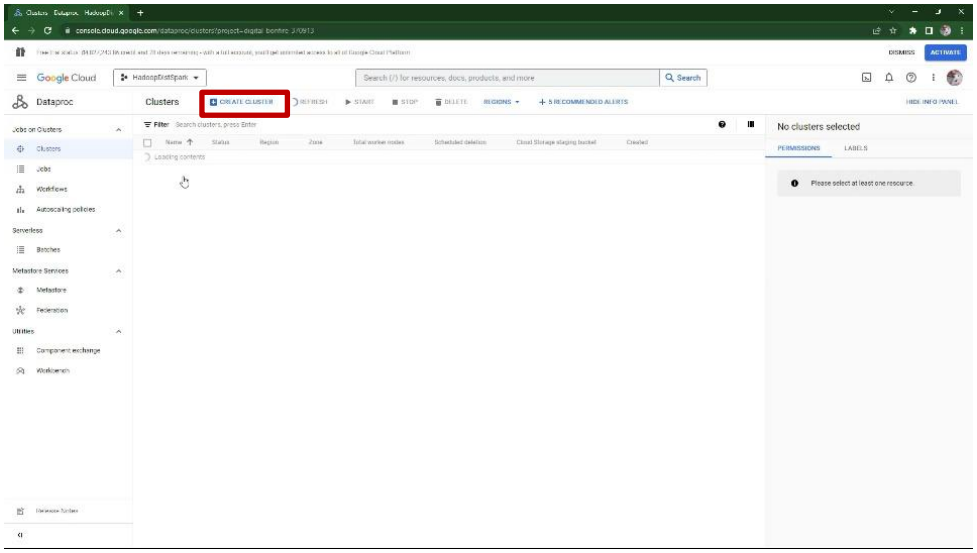
- **Bước 2**: Trong phần khai báo phương thức thanh toán, nhập tài khoản thẻ VISA để kích hoạt và sử dụng được. Google Cloud có cung cấp cho tài khoản mới được sử dụng Free 300$ trong vòng thời gian có hạn.

2.2 Tạo máy ảo Data proc trên Google Cloud

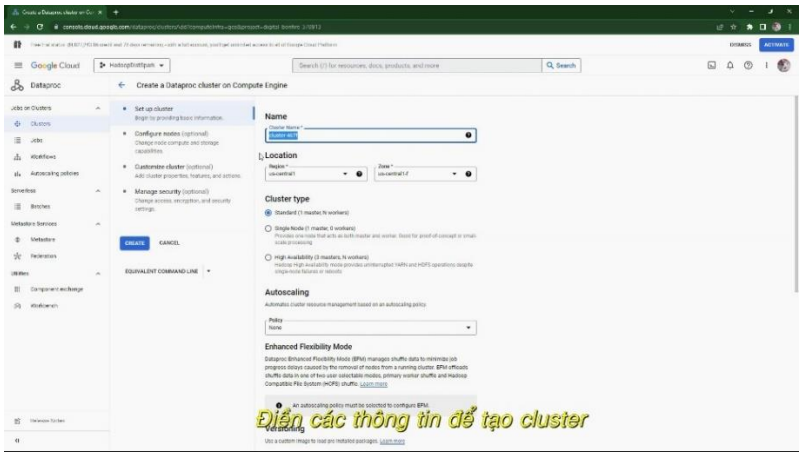
Bước 1: Truy cập vào console.cloud.google.com và chọn Dataproc/ Cluster trên menu bên trái



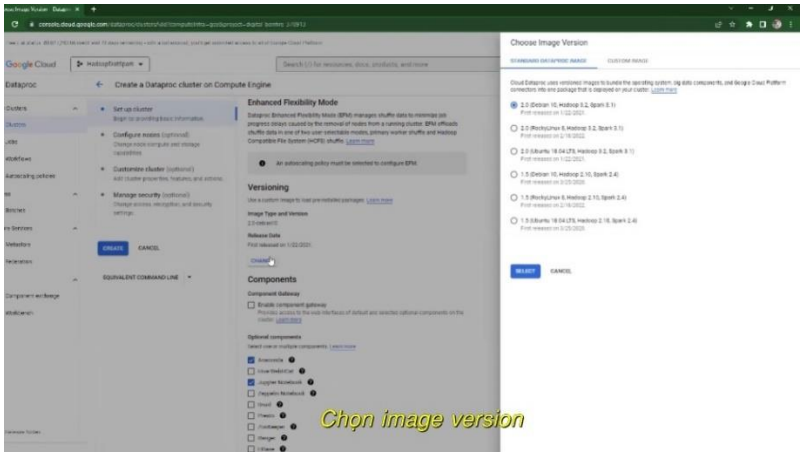
Bước 2: Bấm vào Create Cluster



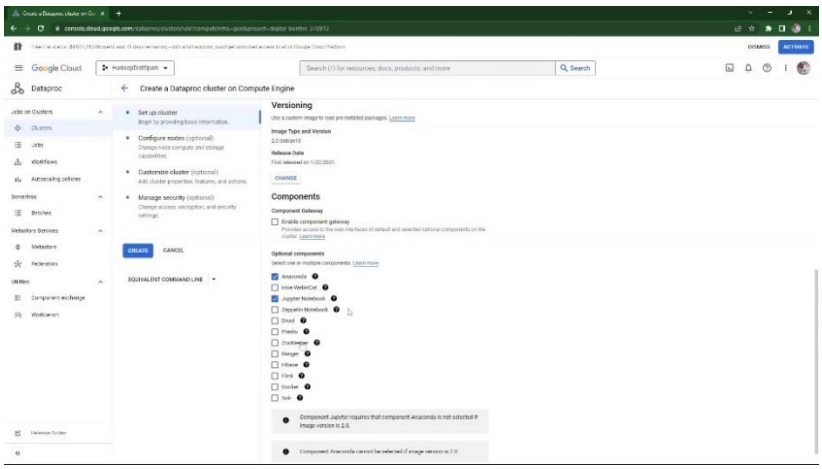
Bước 3: Điền tên cluster



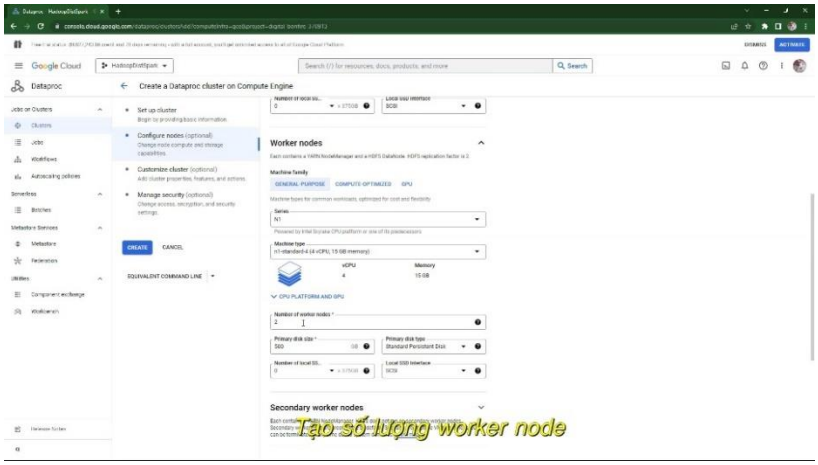
Bước 4: Chọn image version 2.0



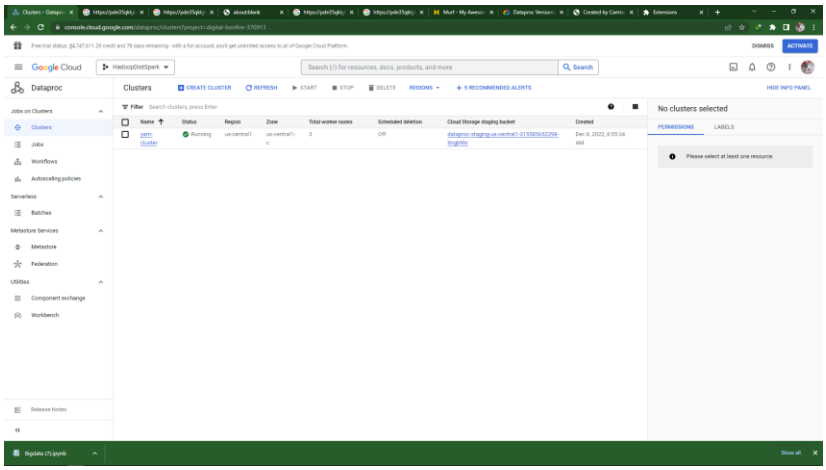
Bước 5: Tích hợp Jupyter Notebook vào Cluster



Bước 6: Tạo số node



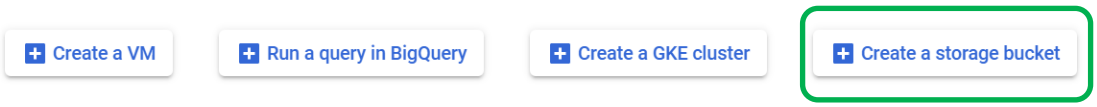
Bước 7: Bấm create



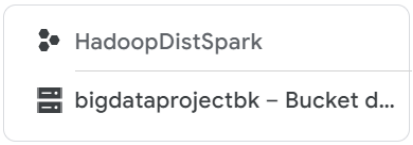
CHƯƠNG 3. XỬ LÝ DỮ LIỆU LỚN

1. LƯU TRỮ DỮ LIỆU

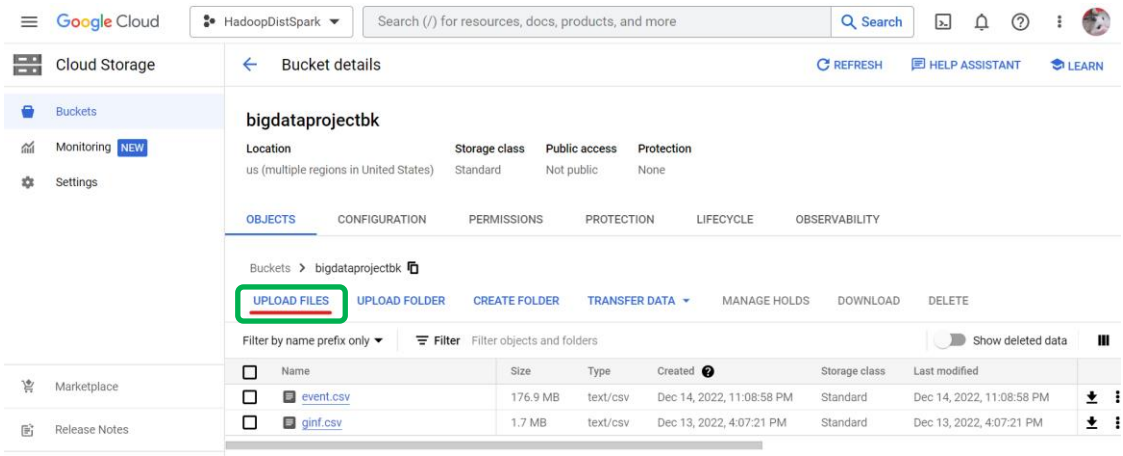
Bước 1: Bấm **Create a storage bucket** để tạo mới một storage



Bước 2: Sau khi tạo ta có



Bước 3: Sau khi tạo xong, nhấn UPLOAD FILES để thêm file csv vào, nó sẽ có đường dẫn sử dụng cho ML như sau: gs://bigdataprojectbk/*.csv



2. TIỀN XỬ LÝ DỮ LIỆU

Bước 1: Thêm một số thư viện cần thiết

```
In [209]: import pandas as pd

          import pyspark.sql.functions as f
          from pyspark.sql.types import *
          from pyspark.sql.functions import udf
          from pyspark.sql.functions import col
          from math import sqrt
          from numpy.linalg import norm
          from pyspark.sql.window import Window
          from pyspark.sql.functions import col, row_number, rank, monotonically_increasing_id
          from operator import add
          from pyspark.sql import Row
```

Bước 2: Đọc file csv cả hai bộ dữ liệu từ Google Cloud Storage

```
In [210]: events_df=spark.read.csv('gs://bigdataprojectbk/event.csv', header=True, inferSchema=True)

In [211]: events_df.printSchema

Out[211]: <bound method DataFrame.printSchema of DataFrame[_c0: int, id_odsp: string, id_event: string, sort_order: int, time: int, text:
          string, event_type: int, event_type2: double, side: int, event_team: string, opponent: string, player: string, player2: string,
          player_in: string, player_out: string, shot_place: double, shot_outcome: double, is_goal: int, location: double, bodypart: doub
          le, assist_method: int, situation: double, fast_break: int, game_id: int]>

In [212]: game_df=spark.read.csv('gs://bigdataprojectbk/ginf.csv', header=True, inferSchema=True)

In [213]: game_df.printSchema

Out[213]: <bound method DataFrame.printSchema of DataFrame[id_odsp: string, link_odsp: string, adv_stats: boolean, date: timestamp, leagu
          e: string, season: int, country: string, ht: string, at: string, fthg: int, ftag: int, odd_h: double, odd_d: double, odd_a: dou
          ble, odd_over: string, odd_under: string, odd_bts: string, odd_bts_n: string]>
```

Bước 3: Khởi tạo hàm chuyển đổi dữ liệu

```
In [214]: def mapKeyToVal(mapping):
              def mapKeyToVal_(col):
                  return mapping.get(col)
              return udf(mapKeyToVal_, StringType())
```

Bước 4: Định nghĩa dữ liệu lại để chuyển đổi

```
In [215]: evtTypeMap = {0:'Announcement', 1:'Attempt', 2:'Corner', 3:'Foul', 4:'Yellow card', 5:'Second yellow card', 6:'Red card', 7:'Subs

          evtTyp2Map = {12:'Key Pass', 13:'Failed through ball', 14:'Sending off', 15:'Own goal', 0:'NA'}

          sideMap = {1:'Home', 2:'Away'}

          shotPlaceMap = {1:'Bit too high', 2:'Blocked', 3:'Bottom left corner', 4:'Bottom right corner', 5:'Centre of the goal', 6:'High a

          shotOutcomeMap = {1:'On target', 2:'Off target', 3:'Blocked', 4:'Hit the bar', 0:'NA'}

          locationMap = {1:'Attacking half', 2:'Defensive half', 3:'Centre of the box', 4:'Left wing', 5:'Right wing', 6:'Difficult angle a

          bodyPartMap = {1:'Right foot', 2:'Left foot', 3:'Head', 0:'NA'}

          assistMethodMap = {0:'None', 1:'Pass', 2:'Cross', 3:'Headed pass', 4:'Through ball', 0:'NA'}

          situationMap = {1:'Open play', 2:'Set piece', 3:'Corner', 4:'Free kick', 0:'NA'}

          countryCodeMap = {'germany':'DEU', 'france':'FRA', 'england':'GBR', 'spain':'ESP', 'italy':'ITA'}

          countryIDMap = {'DEU':'1', 'FRA':'2', 'GBR':'3', 'ESP':'4', 'ITA':'5'}
```

Bước 5: Chuyển đổi dữ liệu cột country và tạo 2 cột mới sau khi chuyển đổi

```
In [216]: game_df = game_df.withColumn("country_code", mapKeyToVal(countryCodeMap)("country"))
          game_df = game_df.withColumn("country_ID", mapKeyToVal(countryIDMap)("country_code"))
```

Bước 6: Chuyển đổi dữ liệu các cột còn lại từ số sang chữ và thêm cột đã chuyển đổi vào bộ dữ liệu, sau đó gom 2 bộ dữ liệu **events** và **ginf** thành một bảng dựa vào thuộc tính chung là **id_odsp**

```
In [217]: events_df = (
                events_df.
                withColumn("event_type_str", mapKeyToVal(evtTypeMap)("event_type")).
                withColumn("event_type2_str", mapKeyToVal(evtTyp2Map)("event_type2")).
                withColumn("side_str", mapKeyToVal(sideMap)("side")).
                withColumn("shot_place_str", mapKeyToVal(shotPlaceMap)("shot_place")).
                withColumn("shot_outcome_str", mapKeyToVal(shotOutcomeMap)("shot_outcome")).
                withColumn("location_str", mapKeyToVal(locationMap)("location")).
                withColumn("bodypart_str", mapKeyToVal(bodyPartMap)("bodypart")).
                withColumn("assist_method_str", mapKeyToVal(assistMethodMap)("assist_method")).
                withColumn("situation_str", mapKeyToVal(situationMap)("situation"))
              )

          joinedDf = (
            events_df.join(game_df, events_df.id_odsp == game_df.id_odsp, 'inner').
            select(events_df.game_id, events_df.id_event, events_df.sort_order, events_df.time, events_df.event_type, events_df.event_type_
          )
```

Bước 7: In bảng sau khi kết hợp để xem

```
In [218]: joinedDf.printSchema

Out[218]: <bound method DataFrame.printSchema of DataFrame[game_id: int, id_event: string, sort_order: int, time: int, event_type: int, e
          vent_type_str: string, event_type2: double, event_type2_str: string, side: int, side_str: string, event_team: string, opponent:
          string, player: string, player2: string, player_in: string, player_out: string, shot_place: double, shot_place_str: string, sho
          t_outcome: double, shot_outcome_str: string, is_goal: int, location: double, location_str: string, bodypart: double, bodypart_s
          tr: string, assist_method: int, assist_method_str: string, situation: double, situation_str: string, country_code: string, coun
          try_ID: string, date: timestamp]>
```

Create time bin for the event

Bước 8: Tạo ngăn cho thuộc tính time để gom nhóm

```
In [219]: from pyspark.ml.feature import QuantileDiscretizer
          joinedDf = QuantileDiscretizer(numBuckets=10, inputCol="time", outputCol="time_bin").fit(joinedDf).transform(joinedDf)
```

Bước 9: Chuyển đổi các cột không đúng định dạng về dạng integer

```
In [220]: joinedDf = joinedDf.withColumn("event_type2",col("event_type2").cast(IntegerType()))
          joinedDf = joinedDf.withColumn("shot_place",col("shot_place").cast(IntegerType()))
          joinedDf = joinedDf.withColumn("shot_outcome",col("shot_outcome").cast(IntegerType()))
          joinedDf = joinedDf.withColumn("location",col("location").cast(IntegerType()))
          joinedDf = joinedDf.withColumn("bodypart",col("bodypart").cast(IntegerType()))
          joinedDf = joinedDf.withColumn("situation",col("situation").cast(IntegerType()))
          joinedDf = joinedDf.withColumn("date",col("date").cast(DateType()))

In [221]: joinedDf.printSchema

Out[221]: <bound method DataFrame.printSchema of DataFrame[game_id: int, id_event: string, sort_order: int, time: int, event_type: int, e
          vent_type_str: string, event_type2: int, event_type2_str: string, side: int, side_str: string, event_team: string, opponent: st
          ring, player: string, player2: string, player_in: string, player_out: string, shot_place: int, shot_place_str: string, shot_out
          come: int, shot_outcome_str: string, is_goal: int, location: int, location_str: string, bodypart: int, bodypart_str: string, as
          sist_method: int, assist_method_str: string, situation: int, situation_str: string, country_code: string, country_ID: string, d
          ate: date, time_bin: double]>
```

Bước 10: Xuất dữ liệu ra để kiểm tra

```
In [222]: joinedDf.limit(3).toPandas()
```

Out[222]:

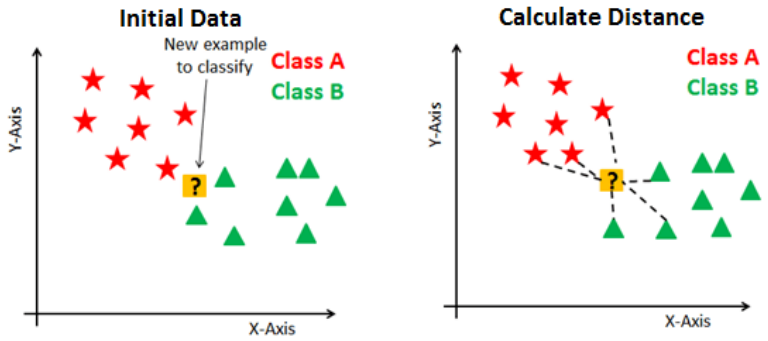
event_type2_str	side	side_str	...	bodypart	bodypart_str	assist_method	assist_method_str	situation	situation_str	country_code	country_ID	date	time_bin
Key Pass	2	Away	...	2	Left foot	1	Pass	1	Open play	DEU	1	2011-08-05	0.0
NA	1	Home	...	0	NA	0	NA	0	NA	DEU	1	2011-08-05	0.0
NA	1	Home	...	0	NA	0	NA	0	NA	DEU	1	2011-08-05	0.0

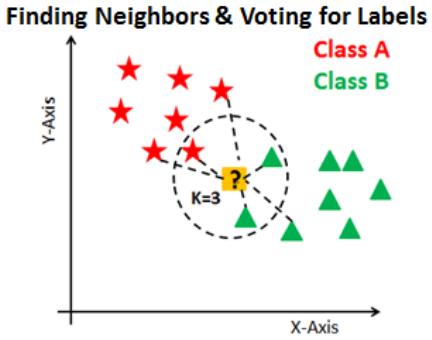
CHƯƠNG 4. GIẢI THUẬT VÀ SONG SONG HÓA GIẢI THUẬT

1. GIỚI THIỆU THUẬT TOÁN K-NEAREST NEIGHBOR

Định nghĩa của thuật toán K-nearest neighbor:

K-nearest neighbor (KNN) là thuật toán đi tìm đầu ra của một điểm dữ liệu mới bằng cách chỉ dựa trên thông tin của K điểm dữ liệu trong training set gần nó nhất (K-lân cận), không quan tâm đến việc có một vài điểm dữ liệu trong những điểm gần nhất này là nhiễu.





Một vài đặc điểm chính của thuật toán KNN:

-K-nearest neighbor là một trong những thuật toán supervised-learning đơn giản nhất, và được vào loại lazy learning.

-K-nearest neighbor có thể áp dụng được vào cả hai loại của bài toán Supervised learning là Classification và Regression.

Cách tính KNN:

Cách 1: Dựa vào khoảng cách(Eucledian Distance)

Ví dụ ta có 2 tập thuộc tính A và B với thuộc tính khác nhau

Ta cần dự đoán thuộc tính bất kì của điểm C(x,y) dựa trên tập các điểm thuộc A và B, thuật toán KNN sẽ tính khoảng cách từ C tới tất cả các điểm trong tập A và B hoặc K điểm lân cận. Nếu K điểm lân cận có số điểm thuộc A hoặc B nhiều hơn thì sẽ quyết định đến thuộc tính của điểm C.

Công thức: $CA=\sqrt{(x1-x2)^2+(y1-y2)^2}$

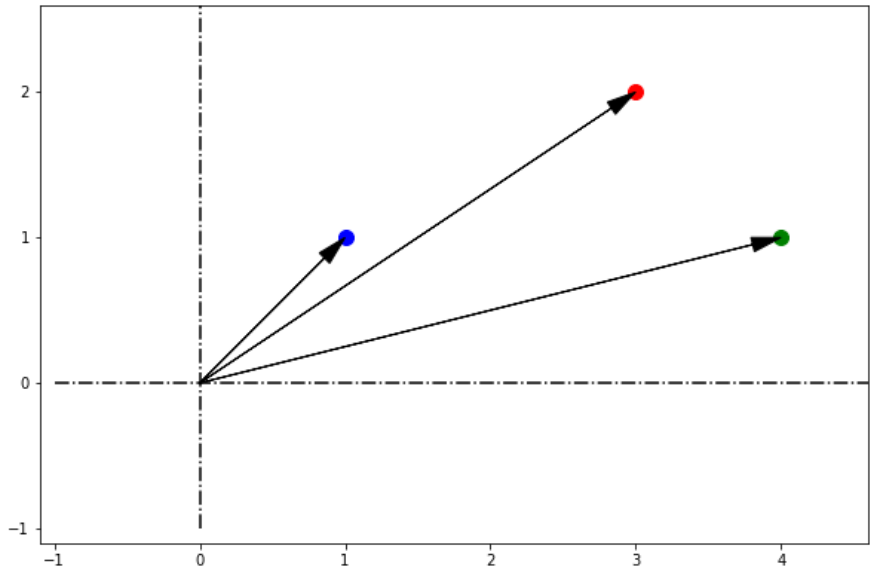
Cách 2: Dựa trên cos của góc ở giữa 2 vector

2 vector A và B trong không gian nhiều chiều bất kì, khi thực hiện phép chiếu xuống 1 mặt phẳng, sẽ hợp với nhau và tạo thành 1 góc α . Ta có thể tính cos của góc đó bằng công thức:

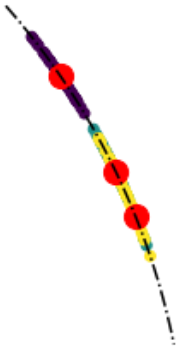
$Cos(\alpha)=\frac{\vec{A}.\vec{B}}{|A|*|B|}$

Trong đó $\vec{A}.\vec{B}$ là tích vô hướng của 2 vector

|A| và |B| là module của 2 vector



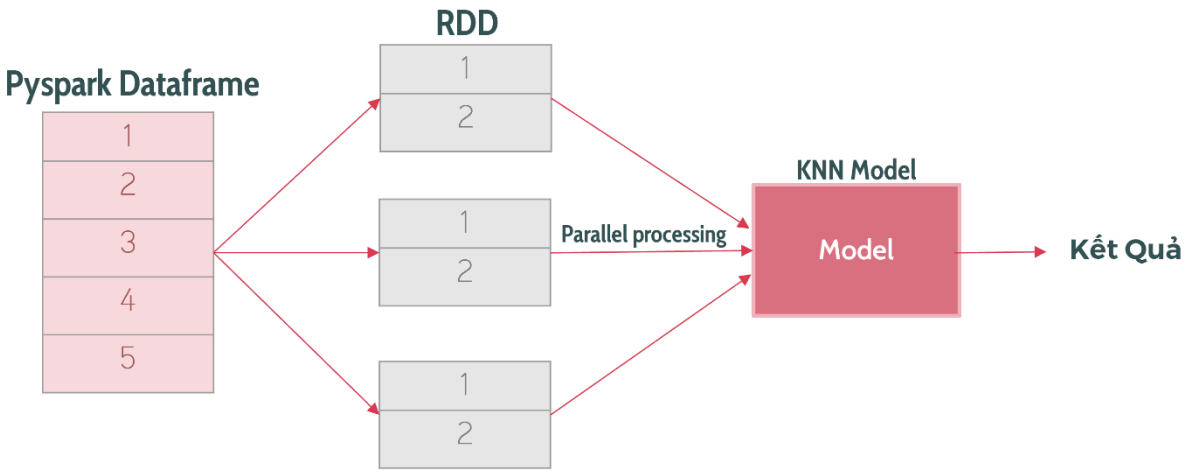
Khi đó, ta không thể xác định được độ gần nhau của các điểm được nữa, thay vào đó, ta phân cụm dữ liệu dựa vào số đo góc



Khi chiếu điểm C xuống theo công thức, nếu điểm C thuộc đường cong của tập A hoặc B thì sẽ quyết định đến thuộc tính của điểm C.

Nhóm chọn cách 2 để thực hiện đề tài.

2. MÔ HÌNH SONG SONG HÓA



3. SỬ DỤNG SQL ĐỂ TRUY XUẤT DỮ LIỆU

Đổi tên bảng để truy xuất dữ liệu

```
In [223]: joinedDf.createOrReplaceTempView("EuropeFootball")
```

1: Xuất ra top 10 tên cầu thủ(player), số lượng bàn thắng(is_goal) của cầu thủ với điều kiện đó là cầu thủ thứ nhất tham gia vào sự kiện

```
In [224]: spark.sql("SELECT player,SUM(is_goal) AS GOAL FROM EuropeFootball GROUP BY player ORDER BY GOAL DESC LIMIT 10").show()
```

```
+--------------------+----+
|              player|GOAL|
+--------------------+----+
|        lionel messi| 205|
|   cristiano ronaldo| 198|
|  zlatan ibrahimovic| 153|
|  robert lewandowski| 124|
|      edinson cavani| 121|
|     gonzalo higuain| 118|
|pierreemerick aub...| 100|
|         luis suarez|  96|
|         diego costa|  93|
| alexandre lacazette|  88|
+--------------------+----+
```

2: Xuất ra top 10 tên cầu thủ(player), số lượng bàn thắng(is_goal) của cầu thủ với điều kiện đó là cầu thủ thứ hai tham gia vào sự kiện

```
In [225]: spark.sql("SELECT player2,SUM(is_goal) AS ASSIST FROM EuropeFootball WHERE player2<>'0' GROUP BY player2 ORDER BY ASSIST DESC LIM
```

```
+-----------------+------+
|          player2|ASSIST|
+-----------------+------+
|     lionel messi|    75|
|   angel di maria|    69|
|   gonzalo castro|    66|
|    dimitri payet|    60|
|    cesc fabregas|    60|
|       mesut ozil|    57|
|     marek hamsik|    56|
|cristiano ronaldo|    53|
|    franck ribery|    51|
|  kevin de bruyne|    51|
+-----------------+------+
```

3: Xuất ra top 10 tên đội, số lượng bàn thắng(is_goal) của đội với điều kiện đó là đội chơi cởi mở

```
In [226]: spark.sql("SELECT event_team,SUM(is_goal) AS GOAL FROM EuropeFootball WHERE situation_str='Open play' GROUP BY event_team ORDER B
```

```
+------------------+----+
|        event_team|GOAL|
+------------------+----+
|         Barcelona| 472|
|       Real Madrid| 429|
|     Bayern Munich| 357|
|Paris Saint-Germain| 327|
|            Napoli| 319|
| Borussia Dortmund| 297|
|          Juventus| 290|
|           AS Roma| 265|
|              Lyon| 246|
|   Atletico Madrid| 243|
+------------------+----+
```

4: Xuất ra top 10 tên đội, số lượng bàn thắng(is_goal) của đội với điều kiện đó là đội đá phạt trực tiếp

```
In [227]: spark.sql("SELECT event_team,SUM(is_goal) AS GOAL FROM EuropeFootball WHERE situation_str='Free kick' OR situation_str='Set piece
```

```
+-------------------+----+
|         event_team|GOAL|
+-------------------+----+
|        Real Madrid|  94|
|Paris Saint-Germain|  88|
|          Barcelona|  80|
|            AS Roma|  73|
|           AC Milan|  71|
|              Lyon|  70|
|           Juventus|  69|
|      Bayern Munich|  66|
|         Fiorentina|  64|
|              Lazio|  63|
+-------------------+----+
```

5: Xuất ra top 10 tên đội, số lượng bàn thắng(is_goal) của đội với điều kiện đó là đội đá góc

```
In [228]: spark.sql("SELECT event_team,SUM(is_goal) AS GOAL FROM EuropeFootball WHERE situation_str='Corner' GROUP BY event_team ORDER BY (
```

```
+-----------------+----+
|       event_team|GOAL|
+-----------------+----+
|  Atletico Madrid|  51|
|      Real Madrid|  50|
|  Athletic Bilbao|  48|
|         Juventus|  42|
|       Fiorentina|  41|
|          Sevilla|  40|
|         AC Milan|  40|
|         Valencia|  40|
|Borussia Dortmund|  40|
|          Bordeaux|  39|
+-----------------+----+
```

6: Xuất ra top 10 tên cầu thủ(player), số lượng bàn thắng(is_goal) của cầu thủ với điều kiện đó là cầu thủ thứ nhất trong sự kiện và ở khoảng cách rất gần khung thành đối phương

```
In [229]: sql("SELECT event_team,SUM(is_goal) AS GOAL FROM EuropeFootball WHERE location_str='Very close range' GROUP BY event_team ORDER B
```

```
+-------------------+----+
|         event_team|GOAL|
+-------------------+----+
|        Real Madrid|  77|
|          Barcelona|  72|
|Paris Saint-Germain|  67|
|      Bayern Munich|  61|
|           Valencia|  58|
|    Atletico Madrid|  56|
|            Sevilla|  54|
|             Napoli|  53|
|     Internazionale|  53|
|              Lazio|  52|
+-------------------+----+
```

7: Xuất ra top 10 tên cầu thủ(player), số lượng bàn thắng(is_goal) của cầu thủ với điều kiện đó là cầu thủ thứ nhất trong sự kiện ở khoảng cách phạt đền

```
In [230]: spark.sql("SELECT player,SUM(is_goal) AS GOAL FROM EuropeFootball WHERE location=14 GROUP BY player ORDER BY GOAL DESC LIMIT 10").
```

```
+------------------+----+
|            player|GOAL|
+------------------+----+
|  cristiano ronaldo|  43|
| zlatan ibrahimovic|  35|
|       lionel messi|  30|
|     edinson cavani|  20|
|             falcao|  20|
|    sejad salihovic|  19|
|alexandre lacazette|  18|
|   antonio candreva|  18|
|        eden hazard|  18|
|   domenico berardi|  16|
+------------------+----+
```

8: Xuất ra top 10 tên cầu thủ(player), số lượng bàn thắng(is_goal) của cầu thủ với điều kiện đó là cầu thủ thứ nhất ở khoảng cách bên cánh trái

```
In [231]: spark.sql("SELECT player,SUM(is_goal) AS GOAL FROM EuropeFootball WHERE situation=4 GROUP BY player ORDER BY GOAL DESC LIMIT 10").
```

```
+-----------------+----+
|           player|GOAL|
+-----------------+----+
|      lionel messi|  14|
|    miralem pjanic|  13|
| cristiano ronaldo|  13|
|      andrea pirlo|  12|
|  hakan calhanoglu|  10|
|    francesco lodi|  10|
|       daniel wass|   9|
|            benat|   9|
|zlatan ibrahimovic|   8|
|       juan arango|   7|
+-----------------+----+
```

9: Xuất ra top 10 tên cầu thủ(player), số lượng bàn thắng(is_goal) của cầu thủ với điều kiện đó là cầu thủ thứ nhất ở sự kiện phản lưới nhà

```
In [232]: spark.sql("SELECT player,SUM(is_goal) AS GOAL FROM EuropeFootball WHERE event_type2=15 GROUP BY player ORDER BY GOAL DESC LIMIT 1
```

```
+----------------+----+
|          player|GOAL|
+----------------+----+
|  gareth mcauley|   4|
|   anthony weber|   4|
|  vangelis moras|   4|
|   martin skrtel|   4|
| aymen abdennour|   3|
|   davide astori|   3|
|     diego mainz|   3|
|shkodran mustafi|   3|
|      harry kane|   3|
|      john terry|   3|
+----------------+----+
```

10: Xuất ra trung bình bàn thắng mỗi trận:

Trung bình số bàn mỗi trận

```
In [69]: spark.sql("SELECT (SUM(is_goal)/COUNT(DISTINCT game_id)) AS AVERAGE_GOAL_PER_GAME FROM EuropeFootball").show()
```

```
+---------------------+
|AVERAGE_GOAL_PER_GAME|
+---------------------+
|   2.6940709720079346|
+---------------------+
```

11: Xuất ra thời gian ghi bàn nhiều nhất

Bàn thắng ở các phút

```
In [62]: spark.sql("SELECT time_bin,SUM(is_goal) AS GOAL FROM EuropeFootball GROUP BY time_bin ORDER BY GOAL DESC LIMIT 10").show()
```

```
+--------+----+
|time_bin|GOAL|
+--------+----+
|     9.0|2688|
|     2.0|2636|
|     4.0|2572|
|     8.0|2541|
|     5.0|2530|
|     6.0|2507|
|     1.0|2331|
|     3.0|2245|
|     7.0|2235|
|     0.0|2161|
+--------+----+
```

12: Bộ phận sử dụng để ghi bàn nhiều nhất

Body part score most goal

```
n [61]: park.sql("SELECT bodypart_str,SUM(is_goal) AS GOAL FROM EuropeFootball GROUP BY bodypart_str ORDER BY GOAL DESC LIMIT 10").show()
```

```
+------------+-----+
|bodypart_str| GOAL|
+------------+-----+
|  Right foot|13451|
|   Left foot| 6758|
|        Head| 4236|
|          NA|    1|
+------------+-----+
```

13: Trận đấu có nhiều bàn nhất

Các trận đấu có nhiều bàn nhất

```
70]: spark.sql("SELECT game_id,SUM(is_goal) AS GOAL FROM EuropeFootball GROUP BY game_id ORDER BY GOAL DESC LIMIT 5").show()
```

```
+-------+----+
|game_id|GOAL|
+-------+----+
|   4657|  12|
|   6862|  11|
|   7393|  10|
|   7353|  10|
|   2053|  10|
+-------+----+
```

```
73]: spark.sql("SELECT event_team, opponent,date FROM EuropeFootball WHERE game_id=4657 LIMIT 1").show()
```

```
+-----------+--------------+----------+
| event_team|      opponent|      date|
+-----------+--------------+----------+
|Real Madrid|Rayo Vallecano|2015-12-20|
+-----------+--------------+----------+
```

4. KHỞI TẠO MỘT SỐ HÀM VÀ XỬ LÝ SONG SONG HÓA

Bước 1: Tính cos của giải thuật KNN

```
In [331]: def cos(a,b):
              s = 0
              for i,j in zip(a,b):
                  s = s + i*j
              return s/(norm(a)*norm(b))
```

Bước 2: Tách cột dữ liệu phù hợp từ bộ dữ liệu gốc của tập dữ liệu train

```
In [324]: def get_game_train_vectors(df, country_code):
              return df.filter(df.country_code == country_code).select(col("feature"), col("is_goal"))
```

Bước 3: Tách cột dữ liệu phù hợp từ bộ dữ liệu gốc của tập dữ liệu test

```
In [325]: def get_game_test_vectors(df, country_code):
              return df.filter(df.country_code == country_code).select(col("feature").alias("test_feature"), col("ActionID").alias("test_id
```

Bước 4: Machine learning(Thuật toán KNN), khởi tạo các giá trị mặc định

```
class KNN_goal_predict:
    def __init__(self, country_code, train_vec, test_vec, k):
        self.country_code = country_code
        self.train_vec = train_vec
        self.test_vec = test_vec
        self.k = k
```

Bước 5: Machine learning(Thuật toán KNN), khởi tạo hàm tính cos và K điểm gần nhất

```
def knn_result(self):
    k = self.k
    tr = self.train_vec
    td = self.test_vec
    cross_join_df = tr.crossJoin(td)
    cosine_mapped_df = cross_join_df\
                .rdd.map(lambda x: (float(cos(x.feature,x.test_feature)), x.is_goal, x.test_id))\
                .toDF(["cos", "is_goal", "test_id"])
    windowDept = Window.partitionBy("test_id").orderBy(col("cos").desc(), col("is_goal").desc())
    top_k_nearest_items_df = cosine_mapped_df\
                            .withColumn("row",row_number().over(windowDept)).filter(col("row") <= k)
    xG_df = top_k_nearest_items_df.rdd.map(lambda x: (x.test_id, x.cos*x.is_goal))\
                                              .reduceByKey(add)\
                                              .map(lambda x: (x[0], (x[1]/k)))\
                                              .toDF(["action_id", "xG"])\
                                              .orderBy(col("xG").desc())\
    return xG_df
```

Bước 6: Machine learning(Thuật toán KNN), Tính Root Mean Square Error

```
def RMSE(self):
    k = self.k
    w = self.train_vec.orderBy(f.rand())
    test_size = int(w.count()/4)
    test_df = w.limit(test_size)
    train_df = w.subtract(test_df)
    test_df = test_df.withColumnRenamed("feature", "test_feature")\
                             .withColumn("test_id", monotonically_increasing_id())
    actual_goal_df = test_df.select(col("test_id"), col("is_goal").alias("actual_goal"))
    cross_join_df = train_df.crossJoin(test_df)
    cosine_mapped_df = cross_join_df\
                .rdd.map(lambda x: (float(cosine(x.feature,x.test_feature)), x.is_goal, x.test_id))\
                .toDF(["cosine_score", "is_goal", "test_id"])
    windowDept = Window.partitionBy("test_id").orderBy(col("cosine_score").desc(), col("is_goal").desc())
    top_k_nearest_items_df = cosine_mapped_df\
                            .withColumn("row",row_number().over(windowDept)).filter(col("row") <= k)
    xG_df = top_k_nearest_items_df.rdd.map(lambda x: (x.test_id, x.cosine_score*x.is_goal))\
                                              .reduceByKey(add)\
                                              .map(lambda x: (x[0], x[1]/k))\
                                              .toDF(["test_id", "xG"])
    join_df = xG_df.join(actual_goal_df, ["test_id"])
    result_rdd = join_df.rdd.map(lambda x: ((x.xG - x.actual_goal)**2))
    return (result_rdd.reduce(add)/result_rdd.count())**(1/2)
```

5. XÂY DỰNG MÔ HÌNH MACHINE LEARNING VỚI SONG SONG HÓA GIẢI THUẬT

Bước 1: Chọn từ bộ dữ liệu những cột mong muốn

```
In [237]: use_cols=joinedDf.select("country_code","shot_place","shot_outcome","location","bodypart","assist_method","situation","is_goal")
```

Bước 2: Sắp xếp dữ liệu phù hợp với model

```
In [238]: actions = use_cols.select("country_code","is_goal", f.array(use_cols.columns[1:]).alias("feature"))
          actions.show(5)
```

```
+------------+-------+--------------------+
|country_code|is_goal|             feature|
+------------+-------+--------------------+
|         DEU|      0|[6, 2, 9, 2, 1, 1...|
|         DEU|      0|[0, 0, 0, 0, 0, 0...|
|         DEU|      0|[0, 0, 0, 0, 0, 0...|
|         DEU|      0|[0, 0, 0, 0, 0, 0...|
|         DEU|      0|[0, 0, 2, 0, 0, 0...|
+------------+-------+--------------------+
only showing top 5 rows
```

Bước 3: Xóa những cột dữ liệu trùng trong cột feature

```
In [239]: actions=actions.dropDuplicates(['feature'])
```

Bước 4: Đặt tên cho bộ dữ liệu để sử dụng query

```
In [240]: actions.createOrReplaceTempView("actions")
```

Bước 5: Tạo cột tự động tăng dần mới trong bộ dữ liệu, và xuất ra dữ liệu để kiểm tra

```
In [241]: action_df=spark.sql("SELECT ROW_NUMBER() OVER(ORDER BY country_code) AS ActionID,country_code,is_goal,feature FROM actions")

In [242]: action_df
Out[242]: DataFrame[ActionID: int, country_code: string, is_goal: int, feature: array<int>]

In [243]: action_df.limit(4).toPandas()
```

Out[243]:

	ActionID	country_code	is_goal	feature
0	1	DEU	0	[6, 2, 7, 3, 2, 3, 0]
1	2	DEU	0	[1, 2, 12, 1, 2, 1, 0]
2	3	DEU	0	[1, 2, 16, 2, 1, 2, 0]
3	4	DEU	1	[4, 1, 9, 1, 4, 1, 1]

Bước 6: Chia tập dữ liệu ra làm train và test, in dữ liệu để kiểm tra

```
In [244]: train, test = action_df.randomSplit([0.9, 0.1])

In [245]: train.take(2)
Out[245]: [Row(ActionID=1, country_code='DEU', is_goal=0, feature=[1, 2, 12, 1, 2, 1, 0]),
           Row(ActionID=2, country_code='DEU', is_goal=0, feature=[1, 2, 16, 2, 1, 2, 0])]
```

Bước 7: Lấy cột dữ liệu feature và is_goal của tập dữ liệu train

```
In [326]: train_vec = get_game_train_vectors(train, 'DEU')
          train_vec.take(1)
Out[326]: [Row(feature=[1, 2, 12, 1, 2, 3, 0], is_goal=0)]
```

Bước 8: Lấy cột dữ liệu feature và ActionID của tập dữ liệu test, đặt tên là test_feature và test_id

```
In [327]: test_vec = get_game_test_vectors(test, 'DEU')
          test_vec.take(1)
Out[327]: [Row(test_feature=[8, 2, 16, 1, 0, 1, 0], test_id=21)]
```

Bước 9: Chạy hàm khởi tạo của thuật toán KNN

```
In [342]: a = KNN_goal_predict("DEU",train_vec,test_vec, 3)
```

Bước 10: Tính RMSE(Root Mean Square Error)

```
In [296]: a.RMSE()
Out[296]: 0.6312690573128753
```

Bước 11: Kết quả sau khi tính toán

```
In [343]: result=a.knn_result().show()
+---------+------------------+
|action_id|                xG|
+---------+------------------+
|      555|0.9998362769260868|
|      281|0.9990749377100562|
|      316|0.9990306816119089|
|      625|0.9989074845677214|
|      585|0.9988576956198494|
|      424|0.9988298297607071|
|      220|0.9985136295161957|
|      746|0.9984437831290339|
|      130| 0.998423707906071|
|      563| 0.998295597691981|
|      299|0.9982011509340611|
|      612|0.9981807722935724|
|      185|0.9981497687902982|
|      459|0.9981435652320368|
|      649|0.9980843293510299|
|      388|0.9979521438808363|
```

CHƯƠNG 5. KẾT QUẢ ĐẠT ĐƯỢC

1. KẾT QUẢ

Lấy ActionID = 281 để thử nghiệm độ chính xác ta được:

```
In [345]: action_df.filter("ActionID==281").select("ActionID","feature").toPandas()
          #"shot_place","shot_outcome","location","bodypart","assist_method","situation","is_goal
```

Out[345]:

	ActionID	feature
0	281	[3, 1, 13, 2, 0, 1, 1]

```
#shot_place: 3 - 'Bottom left corner'
#shot_outcome:  1 - 'On target'
#location: 13 - 'Very close range'
#bodypart: 2 - 'Left foot '
#assis method: 0 - 'None'
#situation: 1 - 'Open play'
#is_goal: 1 - 'yes'
```

Ở tình huống ID 281, cầu thủ thực hiện cú sút vào góc dưới bên trái của khung thành, cú sút đưa bóng đi trúng đích và được thực hiện ở khoảng cách cực kì gần, cú đá được thực hiện bằng chân trái, phương thức kiến tạo = 0 nghĩa là cầu thủ này tự thực hiện 1 pha đi bóng rồi dứt điểm hoặc đây mà một tình huống đá bồi, đây là một pha bóng sống đạt tới 0.99**xG** và thực tế cũng đã trở thành bàn thắng.

2. ĐÁNH GIÁ

Với RMSE =0.6 đã tính ở chương 4, ta có thể nói mô hình dự đoán chưa thật sự chính xác. Điều này gồm có các lí do vừa chủ quan vừa khách quan như:

- Nhóm chưa tìm được phương pháp để tối ưu hóa giá trị K cho thuật toán KNN
- Trong thực tế, để giải quyết một vấn đề bằng machine learning, người ta cần phải sử dụng nhiều mô hình thuật toán khác nhau để so sánh đánh giá, việc sử dụng duy nhất 1 mô hình không thể đem lại độ chính xác cao nhất

- Trong bóng đá, không phải cầu thủ nào cũng có thể tận dụng được 100% các cơ hội 10/10, đôi khi có những tình huống đạt tới 0.99xG nhưng vẫn bị bỏ lỡ đáng tiếc, điều này làm cho tập train không được ổn định.

CHƯƠNG 6. KẾT LUẬN

1. ƯU ĐIỂM

- Hiểu được cách cài đặt mô hình multimode trên google cloud
- Nắm được cơ bản cách hoạt động của Spark và RDD
- Hiểu được cách áp dụng một mô hình Machine Learning trên RDD

2. HẠN CHẾ

- Chưa áp dụng được nhiều thuật toán trong đồ án
- Gặp nhiều khó khăn nên chưa thể triển khai đồ án trên máy local
- Vì chưa thật sự quen với các nền tảng xứ lí dữ liệu lớn nên còn nhiều thiếu sót

3. HƯỚNG PHÁT TRIỂN

- Để mô hình được chính xác hơn, cần sử dụng thêm các thuộc tính đầu vào
- Cần sử dụng nhiều mô hình dự đoán hơn để đưa ra kết quả tốt hơn

CHƯƠNG 7. TÀI LIỆU THAM KHẢO

1. Tìm hiểu về Google Cloud Platform: https://bkhost.vn/blog/gcp-google-cloud-platform/
2. Tìm hiểu về thuật toán KNN: https://machinelearningcoban.com/2017/01/08/knn/#:~:text=M%E1%BB%99t%20c%C3%A1ch%20ng%E1%BA%AFn%20g%E1%BB%8Dn%2C%20KNN,g%E1%BA%A7n%20nh%E1%BA%A5t%20n%C3%A0y%20l%C3%A0%20nhi%E1%BB%85u.
3. Cài Spark trên Ubuntu: https://www.7host.vn/huong-dan-cai-apache-spark-tren-ubuntu-20-04-lts/
4. Spark Parallelize: One of the Most Essential Elements of Spark: https://www.simplilearn.com/tutorials/big-data-tutorial/spark-parallelize
5. How to use Spark clusters for parallel processing Big Data: https://www.freecodecamp.org/news/how-to-use-spark-clusters-for-parallel-processing-big-data-86a22e7f8b50/
6. Machine learning from Scratch : https://github.com/eriklindernoren/ML-From-Scratch
7. Create Spark cluster on GCP https://www.youtube.com/watch?v=mF7s6m3DoYw
8. Data Parallelism VS Model Parallelism https://leimao.github.io/blog/Data-Parallelism-vs-Model-Paralelism/