**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ VIỆN TRÍ TUỆ NHÂN TẠO**

**-----\*\*\*-----**

**BÁO CÁO MÔN HỌC**

**KỸ THUẬT VÀ CÔNG NGHỆ DỮ LIỆU LỚN**

**ĐỀ TÀI**

**NAIVE BAYES & LẬP TRÌNH MAPREDUCE HÓA TRONG PHÂN LỚP VĂN BẢN**

## Nhóm sinh viên thực hiện:

1. Dương Phương Hiểu - 22022659
2. Vũ Trung Hiếu - 22022515
3. Hồ Lê Dương – 22022641
4. Vũ Đức Anh - 22022514

## Giảng viên hướng dẫn:

TS Trần Hồng Việt Ths. Ngô Minh Hương

# HÀ NỘI, 11-12-2024

# MỞ ĐẦU

Trong thời đại bùng nổ thông tin, nơi dữ liệu được sinh ra một cách nhanh chóng với đa dạng các nguồn khác nhau như mạng xã hội, trang web thương mại điện tử, các thiết bị di động,… Với nguồn dữ liệu khổng lồ, đa dạng về kiểu dáng, kích thước và được sinh ra với một tốc độ nhanh chóng, đó là ý niệm sơ khai về dữ liệu lớn (Big Data). Big Data chứa được những giá trị tiềm ẩn vô cùng to lớn, để chúng ta có thể khai tác hiểu rõ hành vi của người dùng, thị trường từ đó có thể đưa ra một chiến lược kinh doanh hiểu quả.

Một trong những ứng dụng quan trọng của Big Data đó là phân loại văn bản (Document Classification). Kỹ thuật này cho phép chúng ta xác định được nhãn dán, phân loại được các tài liệu một cách nhanh chóng thông qua văn bản mà tài liệu nó đang chứa đựng. Phân loại văn bản có ý nghĩa vô cùng to lớn trong thời đại hiện nay từ phân loại hồ sơ ý tê, phân tích xu hướng, nghiên cứu thị trường, gian lận,…

Tuy nhiên, việc xử lý và phân tích dữ liệu lớn, đặc biệt là dữ liệu văn bản, đòi hỏi chúng ta những công cụ và kỹ thuật vô cùng tiên tiến. Điều mà các phương pháp tiếp cận theo hướng truyền thống sẽ gặp khó khắn trong việc xử lý một cách hiệu quả và nhanh chóng với khối lượng dữ liệu khổng lồ này. Do đó, sự kết hợp giữa các kỹ thuật xử lý Big Data như MapReduce, Spark và các thuật toán học máy hiện đại như Naive Bayes, hay là các phương pháp học sâu đầy hiệu quả Neural Network, Long Short-Term Memory đang dần trở thành xu hướng tất yếu

Báo cáo này sẽ tập trung vào việc nghiên cứu và ứng dụng các kỹ thuật tiên tiến trong xử lý Big Data và các phương pháp học máy, học sâu để phân loại văn bản

Báo cáo gồm 4 chương:

Chương 1: Tổng quan về dữ liệu lớn.

Chương 2: Phân loại văn bản với Naive Bayes và MapReduce

Chương 3: Nâng cao hiệu quả phân loại với Naive Bayes và Spark

Chương 4: Kết hợp Neural Network, LSTM và MapReduce

Thông qua báo cáo này, chúng tôi hy vọng sẽ cung cấp cho người đọc một cái nhìn tổng quát về các kỹ thuật tiên tiến trong phân loại tài liệu trên nên tảng của Big Data, đồng thời khả năng ứng dụng to lớn của Big Data và cả của lĩnh vực này trong tương lai.

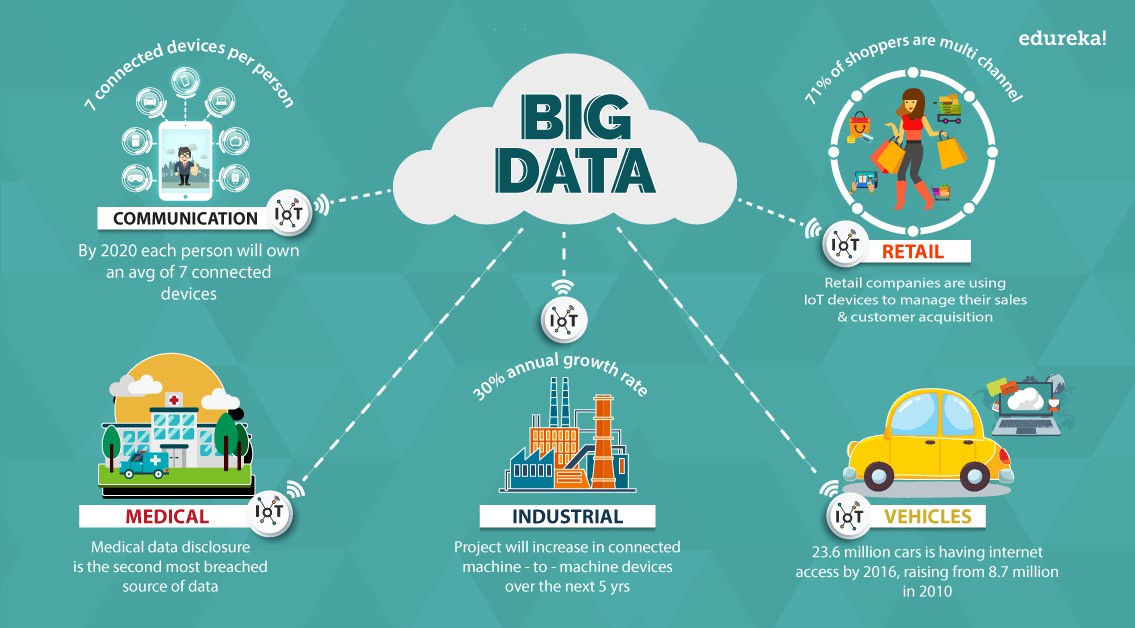
# CHƯƠNG 1: DỮ LIỆU LỚN - NGUỒN LỰC CỦA THỜI ĐẠI SỐ

## Dữ liệu lớn

*Theo Wikipedia:* Dữ liệu lớn là một thuật ngữ chỉ bộ dữ liệu lớn hoặc phức tạp mà các phương pháp truyền thống không đủ các ứng dụng để xử lý dữ liệu này.

*Theo Gartner:* Dữ liệu lớn là những nguồn thông tin có đặc điểm chung khối lượng lớn, tốc độ nhanh và dữ liệu định dạng dưới nhiều hình thức khác nhau, do đó muốn khai thác được đòi hỏi phải có hình thức mới để đưa ra quyết định khám phá và tối ưu hóa quy trình.

Dữ liệu lớn đến từ nhiều nguồn khác nhau:



*Hình 1. Minh họa nguồn gốc của dữ liệu*

Việc khai thác dữ liệu lớn một cách hiệu quả mang lại nhiều lợi ích to lớn cho các tổ chức, cá nhân và xã hội có thể ứng dụng được nó bao gồm: Cắt giảm chi phí, tiết kiệm thời gian, tối ưu hóa sản phẩm, hỗ trợ đưa ra quyết định

Tuy nhiên, việc xử lý và khai thác dữ liệu lớn cũng đặt ra nhiều thách thức về mặt công nghệ, nhân lực và bao gồm cả bảo mật thông tin. Đòi hỏi các tổ chức ứng dụng phải có những công cụ, phương pháp và các hệ thống vô cùng tiên tiến để có thể thu thập, lưu trữ, xử lý và phân tích dữ liệu đó một cách hiệu quả và nhanh chóng.

Mặc dùng còn nhiều thách thức như vậy, nhưng dữ liệu lớn cũng đang mở ra nhiều cơ hội to lớn choc các doanh nghiệp, tổ chức và cá nhân. Ai nắm bắt và khai thắc hiệu quả sẽ là một lợi thế cạnh tranh vô cùng to lớn trong thời đại số.

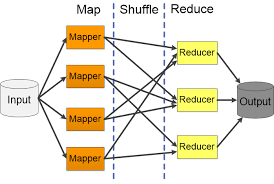
## Đặc trưng cơ bản của dữ liệu lớn

## Big Data, hay là dữ liệu lớn, là thuật ngữ dùng để chỉ những tập dữ liệu có khối lượng khổng lồ, tốc độ phát sinh nhanh và đa dạng về loại hình, đòi hỏi những công nghệ và các phương pháp xử lý mới với các đặc trưng sau:

* + *Khối lượng lớn (Volume):* Dữ liệu lớn có khối lượng vô cùng lớn, vượt xa khả năng lưu trữ và xử lý của các hệ thống truyền thống. Tính đến năm 2014, khối lượng dữ liệu có thể lên đến hàng trăm terabyte, và con số này đang tiếp tục tăng lên theo cấp số nhân.
  + *Tốc độ (Velocity):* Dữ liệu lớn được tạo ra với tốc độ cực nhanh, đòi hỏi khả năng xử lý và phân tích tức thời. Ví dụ, dữ liệu từ mạng xã hội, các giao dịch trực tuyến, cảm biến,... được tạo ra liên tục từng giây, từng phút.
  + *Đa dạng (Variety):* Dữ liệu lớn không chỉ bao gồm dữ liệu có cấu trúc như bảng biểu trong cơ sở dữ liệu, mà còn bao gồm dữ liệu phi cấu trúc như văn bản, hình ảnh, video, âm thanh,... Hiện nay, hơn 80% dữ liệu được tạo ra là dữ liệu phi cấu trúc, đặt ra thách thức lớn cho việc xử lý và phân tích.
  + *Độ tin cậy/ chính xác (Veracity):* Dữ liệu lớn thường chứa đựng nhiều thông tin nhiễu, không chính xác hoặc thiếu nhất quán. Việc đảm bảo độ tin cậy của dữ liệu là một yếu tố quan trọng để đưa ra những kết luận và quyết định chính xác.
  + *Giá trị (Value):* Giá trị của dữ liệu lớn nằm ở khả năng khai thác thông tin hữu ích từ khối lượng dữ liệu khổng lồ này. Thông qua phân tích, chúng ta có thể hiểu rõ hơn về khách hàng, thị trường, xu hướng, từ đó đưa ra những quyết định kinh doanh hiệu quả, tối ưu hóa quy trình và tạo ra sản phẩm dịch vụ tốt hơn.

## Tổng quan về Hadoop

* + MapReduce là một mô hình lập trình xử lý song song và phân tán, được thiết kế để xử lý dữ lý dữ liệu lớn bằng cách chia nhỏ tác vụ và phân phối nó trên nhiều node trong một cụm
  + MapReduce được chia thành hai phần chính, tương ứng với hai pha trong quá trình xử lý dữ liệu:
* Mapper:
* Nhận dữ liệu đầu vào (input splits) từ HDFS.
* Thực hiện xử lý từng phần tử dữ liệu và tạo ra một tập hợp các cặp khóa-giá trị trung gian (<key, value>).
* Reducer:
* Nhận các cặp khóa-giá trị từ Mapper.
* Nhóm các giá trị dựa trên khóa tương ứng.
* Thực hiện các thao tác tổng hợp, xử lý dữ liệu để tạo ra kết quả cuối cùng.
  + Ngoài hai thành phần chính trên, MapReduce còn có các thành phần phụ trợ như:
* **InputFormat:** Xác định cách dữ liệu đầu vào được đọc và chia nhỏ.
* **OutputFormat:** Xác định cách lưu trữ dữ liệu đầu ra.
* **Partitioner:** Chia các cặp khóa-giá trị trung gian từ Mapper vào các phân vùng, tương ứng với các Reducer.
* **Combiner:** Hoạt động giống một Reducer nhỏ gọn, giúp giảm khối lượng dữ liệu trung gian truyền đi.
* **Shuffle and Sort:** Giai đoạn trung gian, đảm bảo dữ liệu được phân phối và sắp xếp trước khi đến Reducer.

*Hình 2. Minh họa* *cấu trúc củaMapReduce*

* + MapReduce hoạt động theo quy trình xử lý dữ liệu song song và phân tán, với hai giai đoạn chính: **Map** và **Reduce**. Dưới đây là chi tiết từng giai đoạn và quy trình tổng thể:
* Giai đoạn Map:
* **Input Splits:** Dữ liệu đầu vào được chia thành các phần nhỏ hơn (input splits) để xử lý đồng thời trên nhiều node.
* **Mapper:** Các Mapper nhận input splits, xử lý chúng để tạo ra các cặp khóa-giá trị trung gian (<key, value>).
* **Intermediate Data:** Kết quả trung gian được gửi đến giai đoạn Shuffle and Sort.
* Giai đoạn Shuffle and Sort:
* **Shuffle:** Dữ liệu trung gian từ Mapper được chuyển đến các Reducer tương ứng.
* **Sort:** Dữ liệu trung gian được sắp xếp theo khóa. Điều này giúp các Reducer dễ dàng xử lý và tổng hợp dữ liệu.
* Giai đoạn Reduce:
* **Reducer:** Các Reducer nhận các cặp khóa-giá trị đã được nhóm lại và thực hiện tổng hợp hoặc xử lý cuối cùng để tạo ra đầu ra.
* **Output:** Kết quả cuối cùng được ghi lại trên hệ thống tệp phân tán (HDFS).
* Tổng thể quá trình:
* Quá trình diễn ra trên một cụm máy tính, mỗi node thực hiện một phần công việc của Mapper hoặc Reducer.
* Quản lý tài nguyên, phân công nhiệm vụ và xử lý lỗi được thực hiện bởi hệ thống quản lý (như YARN).

MapReduce là một trong những bước tiến lớn giúp khai thác sức mạnh của Big Data. Nó giúp các doanh nghiệp và tổ chức tận dụng được giá trị ẩn trong dữ liệu lớn thông qua khả năng xử lý dữ liệu hiệu quả, chi phí thấp và đáng tin cậy. Mặc dù có những hạn chế như tốc độ chậm trong một số trường hợp, MapReduce vẫn là nền tảng của nhiều hệ thống xử lý dữ liệu lớn hiện nay, đặc biệt khi tích hợp với các công nghệ hiện đại như Apache Spark.

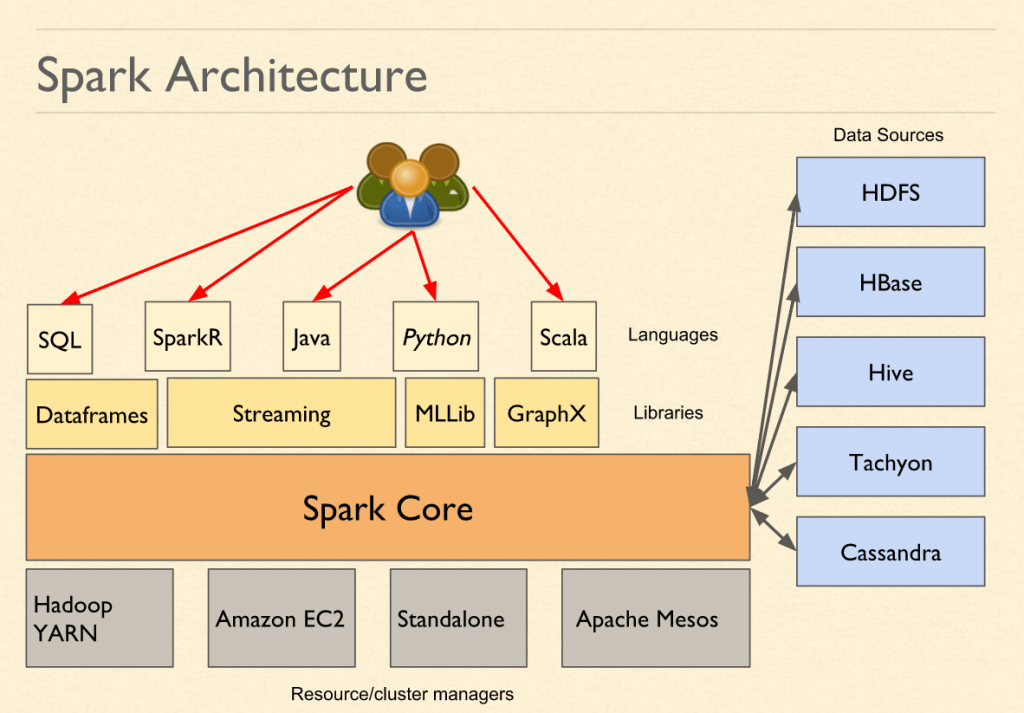
## Tổng quan về PySpark

* + Apache Spark là một framework xử lý dữ liệu lớn, được thiết kế để tối ưu hóa tốc độ và khả năng mở rộng khi xử lý các tác vụ tính toán phức tạp. Spark được phát triển bởi AMPLab tại Đại học California, Berkeley, và hiện là một trong những công nghệ chủ đạo trong hệ sinh thái Big Data.
  + Spark được thiết kế để giải iải quyết những hạn chế về tốc độ và tính linh hoạt của MapReduce, đồng thời được cải thiện tốc độ nhờ xử lý trong bộ nhớ (in-memory), hỗ trợ nhiều mô-đun tính toán, và tích hợp với các công nghệ Big Data khác như Hadoop, HDFS. Ngoài ra Spark là một mô hình rất mạnh mẽ được ứng dụng rộng rãi sử dụng rộng rãi trong phân tích dữ liệu, học máy (machine learning), xử lý luồng dữ liệu (streaming), và các tác vụ ETL (Extract, Transform, Load).



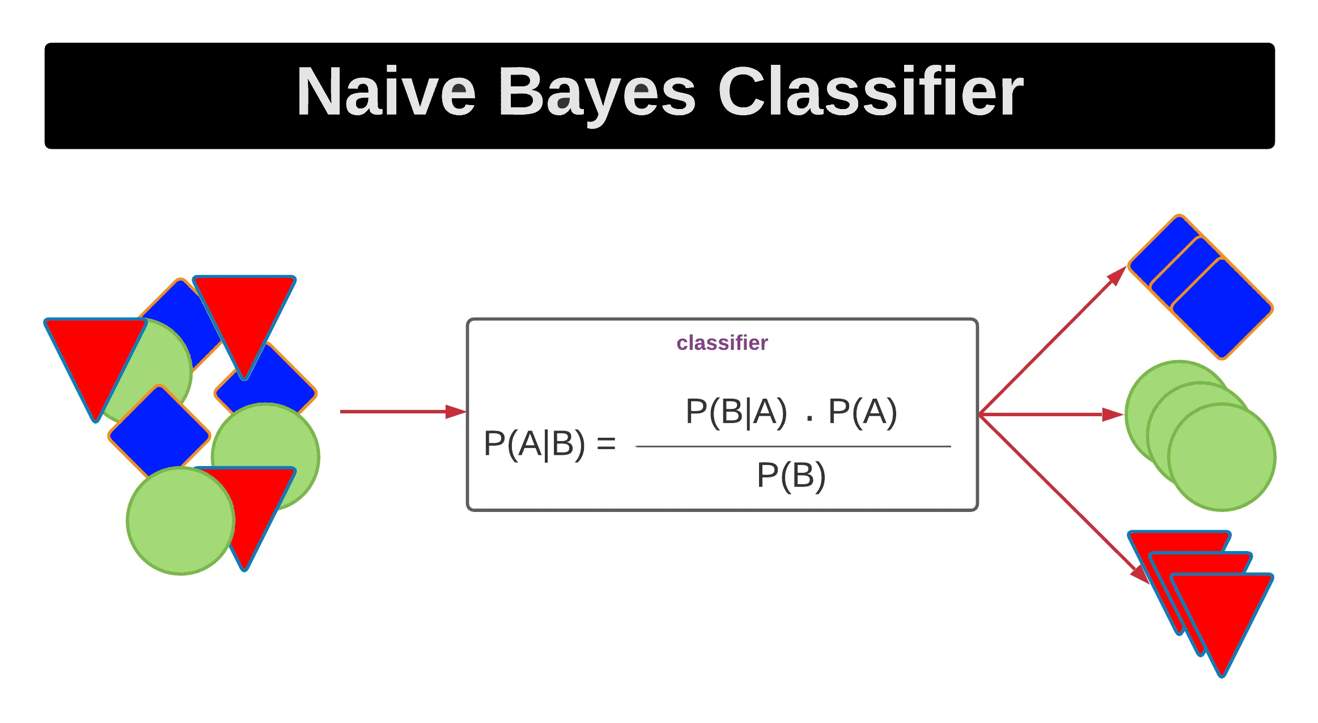
*Hình 4. Spark so với Mapreduce*

* + Spark có kiến trúc phân lớp bao gồm nhiều thành phần chính:
* Spark Core:
* Lớp trung tâm của Spark, chịu trách nhiệm thực hiện các tác vụ cơ bản như lập lịch công việc, quản lý bộ nhớ, và xử lý lỗi.
* Hỗ trợ các API để thao tác với dữ liệu phân tán thông qua Resilient Distributed Datasets (RDDs).
* Spark SQL:
* Mô-đun cho phép người dùng truy vấn dữ liệu dạng cấu trúc bằng ngôn ngữ SQL hoặc giao diện DataFrame.
* Tích hợp với các hệ thống dữ liệu lớn như Hive và các cơ sở dữ liệu quan hệ khác.
* Spark Streaming:
* Xử lý luồng dữ liệu thời gian thực hoặc gần thời gian thực.
* Có khả năng phân tích dữ liệu luồng từ các nguồn như Kafka, Flume.
* MLlib (Machine Learning Library):
* Thư viện hỗ trợ các thuật toán học máy như phân loại, hồi quy, clustering, và xử lý đồ thị.
* GraphX:
* Mô-đun dành cho phân tích đồ thị (graph processing), hỗ trợ các thuật toán như PageRank, Connected Components.
* SparkR và PySpark:
* Cung cấp API để tích hợp Spark với R và Python, giúp các nhà phân tích dữ liệu dễ dàng sử dụng Spark.

 *Hình 5. Kiến trúc của Spark*

* + Spark có kiến trúc phân lớp bao gồm nhiều thành phần chính:
* Spark Core:
* Lớp trung tâm của Spark, chịu trách nhiệm thực hiện các tác vụ cơ bản như lập lịch công việc, quản lý bộ nhớ, và xử lý lỗi.
* Resilient Distributed Datasets (RDDs):
* Là cấu trúc dữ liệu lõi, hỗ trợ lưu trữ và xử lý dữ liệu phân tán.
* RDD có khả năng chịu lỗi nhờ cơ chế lineage, giúp tái tạo dữ liệu bị mất.
* Xử lý trong bộ nhớ (In-memory Computing):
* Dữ liệu được giữ trong bộ nhớ RAM thay vì đọc/ghi từ đĩa, giúp tăng tốc độ xử lý đáng kể.
* Quy trình làm việc:
* **Step 1:** Đọc dữ liệu từ các nguồn như HDFS, S3, hoặc Kafka.
* **Step 2:** Chuyển đổi dữ liệu (transformations) như map, filter để tạo ra các RDD mới.
* **Step 3:** Thực thi các hành động (actions) như reduce, collect để tính toán và lưu trữ kết quả.
* **Step 4:** Quản lý tài nguyên và phân phối công việc thông qua Spark Cluster Manager (Standalone, Mesos, hoặc YARN).
  + Apache Spark đã trở thành một công nghệ không thể thiếu trong lĩnh vực Big Data. Với tốc độ, tính linh hoạt, và khả năng xử lý toàn diện, Spark không chỉ đáp ứng nhu cầu hiện tại mà còn mở ra nhiều cơ hội để khai thác giá trị tiềm năng từ dữ liệu lớn.

## Tổng quan về Naive Bayes

* + Naive Bayes là một thuật toán học máy dựa trên định lý Bayes và giả định độc lập giữa các đặc trưng. Đây là một trong những thuật toán phân loại đơn giản nhưng hiệu quả, được sử dụng rộng rãi trong phân loại văn bản, lọc email spam, và phân tích cảm xúc. Naive Bayes giúp quyết các bài toán phân loại với dữ liệu lớn và đa dạng. Nổi bật với sự đơn giản, dễ triển khai và hiệu quả trên nhiều loại dữ liệu.
  + Naive Bayes dựa trên mô hình xác suất với các thành phần chính:
* Định lý Bayes:
* Xác suất của một lớp (label) CCC dựa trên các đặc trưng (features) XXX được tính theo công thức:
* Với giả định độc lập giữa các đặc trưng, công thức này được đơn giản hóa.
* Giả định độc lập:
* Thuật toán giả định rằng các đặc trưng trong dữ liệu là độc lập với nhau, giúp giảm độ phức tạp tính toán
* Các biến thể của Naive Bayes:
* **Multinomial Naive Bayes:** Phù hợp cho dữ liệu dạng tần suất (như số lần xuất hiện từ trong văn bản).
* **Bernoulli Naive Bayes:** Phù hợp cho dữ liệu nhị phân (như có hoặc không có một từ cụ thể).
* **Gaussian Naive Bayes:** Phù hợp cho dữ liệu liên tục, giả định các đặc trưng có phân phối chuẩn.
  + Naive Bayes hoạt động theo quy trình đơn giản nhưng hiệu quả:
* Giai đoạn huấn luyện:
* Tính toán xác suất tiên nghiệm P(C) cho từng lớp dựa trên dữ liệu huấn luyện.
* Tính toán xác suất có điều kiện P(X∣C) cho từng đặc trưng và lớp.
* Giai đoạn dự đoán:
* Với một dữ liệu mới, áp dụng định lý Bayes để tính xác suất hậu nghiệm P(C∣X) cho từng lớp.

*Hình 6. Quá trình hoạt động của Naive Bayes*

* Chọn lớp có xác suất hậu nghiệm cao nhất làm kết quả phân loại.
  + Naive Bayes là một thuật toán mạnh mẽ và hiệu quả trong xử lý Big Data, đặc biệt khi kết hợp với các công cụ hiện đại như Hadoop hoặc Spark. Mặc dù có những hạn chế, tính đơn giản và khả năng mở rộng của Naive Bayes khiến nó trở thành lựa chọn phù hợp cho nhiều bài toán phân loại trong thế giới dữ liệu lớn.

## Tổng quan về Neural Network

## Neural Network (mạng nơ-ron nhân tạo) là một mô hình tính toán dựa trên cách hoạt động của não bộ con người. Nó được thiết kế để học hỏi từ dữ liệu và có khả năng thực hiện các tác vụ như phân loại, dự đoán, và nhận dạng mẫu. Neural Network là nền tảng cho các phương pháp học sâu (Deep Learning) hiện đại. Neural Network giải quyết các bài toán học máy phức tạp mà các phương pháp truyền thống gặp khó khăn, có khả năng học từ dữ liệu lớn, phi cấu trúc và biểu diễn các mối quan hệ phức tạp.

## Neural Network bao gồm các thành phần chính sau đây:

## Neuron:

## Đơn vị cơ bản của mạng, thực hiện các phép tính và truyền tín hiệu.

## Mỗi neuron nhận dữ liệu từ đầu vào, áp dụng trọng số (weight) và hàm kích hoạt (activation function) để tạo ra đầu ra.

## Các lớp (Layers):

## **Input Layer:** Nhận dữ liệu đầu vào từ tập dữ liệu.

## **Hidden Layers:** Xử lý dữ liệu, mỗi lớp có thể học các đặc trưng phức tạp hơn từ dữ liệu.

## **Output Layer:** Tạo ra kết quả đầu ra, ví dụ như dự đoán hoặc phân loại.

## Trọng số và Bias:

## Trọng số quyết định mức độ quan trọng của các đầu vào, còn bias giúp điều chỉnh giá trị ngưỡng cho mỗi neuron.

## Hàm kích hoạt:

## Hàm phi tuyến được áp dụng tại mỗi neuron, giúp mô hình học các mối quan hệ phức tạp. Ví dụ: Sigmoid, ReLU, Tanh.

## 

*Hình 7. Cấu trúc của neural network*

## Quy trình hoạt động của Neural Network bao gồm:

## Lan truyền tiến (Forward Propagation):

## Dữ liệu đầu vào được truyền qua các lớp trong mạng, từ lớp đầu vào qua các lớp ẩn đến lớp đầu ra.

## Tại mỗi lớp, dữ liệu được biến đổi thông qua trọng số, bias, và hàm kích hoạt.

## Lan truyền ngược (Backward Propagation):

## Dựa trên lỗi (error) giữa đầu ra dự đoán và giá trị thực tế, mạng điều chỉnh các trọng số thông qua thuật toán Gradient Descent.

## Huấn luyện:

## Neural Network học bằng cách lặp đi lặp lại quá trình lan truyền tiến và lan truyền ngược trên tập dữ liệu huấn luyện.

## Dự đoán:

## Sau khi huấn luyện, mô hình có thể dự đoán đầu ra cho dữ liệu mới bằng cách lan truyền tiến duy nhất.

## Neural Network là một công cụ mạnh mẽ trong lĩnh vực Big Data, cho phép khai thác giá trị từ các tập dữ liệu lớn và phức tạp. Khi được kết hợp với các hệ thống xử lý phân tán như Spark hoặc Hadoop, Neural Network có thể xử lý khối lượng dữ liệu khổng lồ với hiệu quả cao, mở ra nhiều cơ hội ứng dụng trong các lĩnh vực công nghệ và kinh doanh.

## Tổng quan về Long Short-Term Memory

## LSTM là một loại mạng nơ-ron hồi tiếp (RNN) được đưa ra lần đầu tiên bởi Hochreiter và Schmidhuber vào năm 1997. LSTM đã khắc phục được những hạn chế của RNN truyền thống, đặc biệt là khả năng học từ những chuỗi dài và nhớ lại thông tin quan trọng từ quá khứ lâu dài. Điều này khiến LSTM rất hữu ích trong các bài toán liên quan đến dữ liệu chuỗi hoặc thời gian, chẳng hạn như dự đoán chuỗi, phân tích văn bản, dịch máy, và nhận dạng giọng nói.

## LSTM có một cấu trúc đặc biệt bao gồm ba cổng chính giúp quản lý dòng thông tin qua mạng nơ-ron:

## **Cổng quên (Forget Gate)**: Quyết định phần nào của trạng thái nhớ cũ cần được giữ lại và phần nào cần bị quên đi.

## **Cổng đầu vào (Input Gate)**: Kiểm soát thông tin mới nào sẽ được ghi vào trạng thái nhớ.

## **Cổng đầu ra (Output Gate)**: Quyết định thông tin nào sẽ được xuất ra từ mô hình tại mỗi bước thời gian.

## Các cổng này giúp LSTM lưu trữ và truy xuất thông tin qua nhiều bước thời gian mà không bị mất dần thông tin như RNN truyền thống.

## *Hình 7. Cấu trúc của mạng LSTM*

* + LSTM hoạt động theo các bước sau:
* **Cổng quên (Forget Gate)**: Dựa trên đầu vào và trạng thái ẩn của thời điểm trước, cổng này quyết định những phần nào của trạng thái nhớ cũ cần được bỏ đi. Điều này giúp loại bỏ những thông tin không cần thiết.
* **Cổng đầu vào (Input Gate)**: Cổng này sẽ quyết định thông tin mới nào sẽ được thêm vào trạng thái nhớ của tế bào LSTM. Thông tin mới được tạo ra từ hai thành phần: đầu vào hiện tại và trạng thái ẩn của tế bào.
* **Cập nhật trạng thái nhớ (Cell State Update)**: Trạng thái nhớ của tế bào được cập nhật bằng cách kết hợp thông tin mới từ cổng đầu vào và thông tin cũ mà cổng quên không loại bỏ.
* **Cổng đầu ra (Output Gate)**: Cổng này quyết định phần nào của trạng thái nhớ sẽ được xuất ra làm đầu ra của tế bào LSTM tại thời điểm hiện tại.
  + Thông qua việc sử dụng các cổng này, LSTM có thể học được thông tin lâu dài mà không bị mất dần như các mô hình RNN thông thường.
  + LSTM đóng vai trò quan trọng trong việc xử lý dữ liệu chuỗi dài, giúp giải quyết các vấn đề như dự đoán chuỗi thời gian, phân tích văn bản và nhận dạng giọng nói. Trong Big Data, LSTM có khả năng học từ các dữ liệu phức tạp và khối lượng lớn, hỗ trợ các ứng dụng trong tài chính, y tế, và thương mại điện tử. Nhờ khả năng lưu trữ thông tin dài hạn và xử lý dữ liệu chuỗi, LSTM là công cụ mạnh mẽ trong phân tích và dự đoán các xu hướng trong các hệ thống Big Data.

# CHƯƠNG 2: PHÂN LOẠI VĂN BẢN VỚI NAIVE BAYES VÀ MAPREDUCE

## Giới thiệu về phân loại văn bản (document classification)

* *Giới thiệu*: Phân loại văn bản là quá trình tự động phân loại các văn bản thành các nhóm hoặc lớp theo một tiêu chí nhất định. Đây là một bài toán phổ biến trong lĩnh vực xử lý ngôn ngữ tự nhiên (NLP) và có ứng dụng rộng rãi trong nhiều lĩnh vực như phân loại email (spam hay không spam), phân tích cảm xúc trong các bài viết, phân loại tin tức theo chủ đề, và nhiều ứng dụng khác. Phân loại văn bản giúp tự động hóa việc xử lý và quản lý lượng lớn dữ liệu văn bản, giúp tiết kiệm thời gian và nguồn lực so với phương pháp thủ công.
* *Ý tưởng*: Ý tưởng của phân loại văn bản là sử dụng các đặc trưng của văn bản (như từ vựng, tần suất xuất hiện của từ, hay các mô hình thống kê) để phân nhóm các văn bản vào các lớp khác nhau. Các thuật toán học máy, như Naive Bayes, là những công cụ phổ biến để thực hiện phân loại văn bản. Naive Bayes dựa trên lý thuyết xác suất và giả định rằng các từ trong văn bản là độc lập với nhau, mặc dù trong thực tế, điều này không hoàn toàn chính xác, nhưng thuật toán này vẫn cho kết quả khá hiệu quả. Khi làm việc với tập dữ liệu lớn (Big Data), việc xử lý và huấn luyện mô hình phân loại trở nên phức tạp. Để giải quyết vấn đề này, **MapReduce** được sử dụng như một công cụ phân tán để xử lý dữ liệu, chia nhỏ công việc thành các phần nhỏ và xử lý song song trên nhiều máy tính. MapReduce giúp tăng tốc quá trình huấn luyện mô hình phân loại văn bản khi phải xử lý lượng dữ liệu lớn, giúp tiết kiệm tài nguyên và thời gian.

## Triển khai bài toán phân tích cảm xúc

## Mô tả bài toán

*Input:* Bộ dữ liệu gồm văn bản của các tài liệu cần được phân loại, mỗi văn bản của mỗi tài liệu sẽ có cột label (dạng số) tương ứng với label mà tài liệu thuộc về, ngoài ra sẽ có cột text\_label (dạng text) tương ứng với label bằng chữ tương ứng với label dạng số

*Output:* Phân loại văn bản trong các tài kiệu được phân loại thành các label tương ứng dạng số, hoặc dạng chữ

## Các bước triển khai

## Bước 1: Đếm tần suất của từ và từ-nhãn (Word Count)

## **Mapper (**mapper\_word\_count**)**:

## Xử lý mỗi dòng dữ liệu, tách thành các thành phần (text, label), và đếm tần suất xuất hiện của từ, từ-nhãn và nhãn.

## Trả về các giá trị word, label, và cặp word-label để giảm (reduce)

## **Reducer (**reducer\_word\_count**)**:

## Tính tổng tần suất của từng từ, từ-nhãn và nhãn.

## Các giá trị tần suất này sẽ được sử dụng trong các bước tính toán tiếp.

## Bước 2: Tính toán xác suất Naive Bayes

## **Mapper (**mapper\_naive\_bayes**)**:

## Truyền dữ liệu từ bước 1 tới bước tiếp theo mà không thay đổi nội dung.

## **Reducer (**reducer\_naive\_bayes**)**:

## Tính toán các xác suất cần thiết:

## **P(C)**: Xác suất của nhãn.

## **P(X)**: Tần suất xuất hiện của từ.

## **P(X|C)**: Xác suất của từ trong mỗi nhãn.

## Dùng phép điều chỉnh Laplace để tránh vấn đề xác suất bằng 0 cho các từ chưa gặp.

## Lưu các xác suất này vào các cấu trúc dữ liệu như word\_label\_probs, label\_probs.

## Bước 3: Lưu mô hình

## Reducer cuối (reducer\_final\_naive\_bayes):

## Lưu mô hình Naive Bayes (bao gồm các xác suất và thông tin liên quan) vào file để sử dụng trong các dự đoán sau này.

## Ví dụ minh họa

## Dữ liệu huấn luyện của chúng ta sẽ bao gồm các tài liệu với các từ và nhãn (Spam hoặc Ham). Ví dụ dữ liệu có thể trông như sau:

|  |  |
| --- | --- |
| Text | Label |
| Free money now | Spam |
| Meeting at 3 PM | Ham |
| Cheap pills | Spam |
| Lunch with friends | Ham |

## Từng tài liệu sẽ được xử lý để tách ra các từ, và mô hình sẽ tính toán các xác suất dựa trên các từ này để phân loại email mới.

## Triển khai MapReduce cho Naive Bayes

## Bước 1: Đếm tần suất của từ và từ-nhãn (Word Count)

## **Mapper (mapper\_word\_count)**:

## Lấy mỗi dòng dữ liệu (email và nhãn) và đếm số lần xuất hiện của từ, từ-nhãn và nhãn.

## Các từ sẽ được đếm ở dạng word, nhãn sẽ được đếm ở dạng label, và cặp từ-nhãn sẽ được đếm dưới dạng word-label.

## Example: ("Free", 1), ("money", 1), ("now", 1), ("Spam", 1)

## ("Meeting", 1), ("at", 1), ("3", 1), ("PM", 1), ("Ham", 1)

## **Reducer (reducer\_word\_count)**:

## Tính tổng tần suất của từng từ, từ-nhãn và nhãn.

## Example: ("Free", 1), ("money", 1), ("now", 1), ("Spam", 2)

## ("Meeting", 1), ("at", 1), ("3", 1), ("PM", 1), ("Ham", 2)

## Bước 2: Tính toán xác suất Naive Bayes

## **Mapper (mapper\_naive\_bayes)**:

## Truyền dữ liệu từ bước 1 (tần suất từ và nhãn) tới bước tiếp theo mà không thay đổi.

## **Reducer (reducer\_naive\_bayes)**:

## Tính toán các xác suất cần thiết

## Dùng Laplace smoothing để tránh xác suất bằng 0 cho các từ chưa xuất hiện trong tập huấn luyện.

## Example:

## P(Spam) = 2/4 = 0.5

## P(Ham) = 2/4 = 0.5

## P("Free"|Spam) = 1/3 = 0.33

## P("money"|Spam) = 1/3 = 0.33

## P("Meeting"|Ham) = 1/3 = 0.33

## P("at"|Ham) = 1/3 = 0.33

## Bước 3: Lưu mô hình

## Ý tưởng MapReduce

* *Nhiệm vụ*: Sử dụng MapReduce để phân loại tài liệu theo label của các tập dữ liệu tài liệu lớn.
* *Ý tưởng*:
* **Mapper**: Xử lý từng tài liệu (document), tách thành các từ và gán nhãn (label) tương ứng với tài liệu đó. Trả về các cặp key-value (word, label) để đếm số lần xuất hiện của mỗi từ trong từng nhãn.
* Reducer: Tính toán tần suất các từ cho từng nhãn, tính toán xác suất P(word|label) và P(label). Các xác suất này sẽ được sử dụng để phân loại tài liệu mới dựa trên từ vựng của chúng..

## Giải pháp MapReduce hóa phân loại tài liệu

* + 1. Dữ liệu đầu vào:

+ Input.txt: Chứa thông tin các bình luận – dữ liệu này được chuyển sang kiểu key/value làm đầu vào cho thuật toán.

+ Keyword.txt: Chứa từ khóa

+ Positive.txt: Chứa các từ mang ý nghĩa tích cực

+ Negative.txt: Chứa các từ mang ý nghĩa tiêu cực

* + 1. Triển khai:

+ Biểu diễn dữ liệu: Dữ liệu được lưu trữ dưới dạng list các hàng văn bản trong file input.txt, mỗi hàng bao gồm một id và một text.

+ Lưu trữ phân tán dữ liệu

+ Trong mỗi vòng lặp

* + - * Mapper: Đọc vào từng dòng dữ liệu từ file input.txt, tách id và text sau đó chuyển text thành key và id thành value.
      * Reducer: Xác định từ khóa từ file “keyword.txt”, kiểm tra sự xuất hiện của từ khóa trong từng bình luận và đếm số lượng từ mang ý nghĩa tích cực và tiêu cực trong bình luận.
      * Kết quả: Tính toán và xác định trạng thái cảm xúc của từng bình luận (tích cực, tiêu cực hoặc trung tính)
    1. Mô hình cơ bản của MapReduce:

+ Mapper (keyIn, valIn) -> list (keyInt, valInt)

+ Reducer (keyInt, valInt) -> list (keyOut, valOut)

* + 1. Áp dụng cho việc phân tích cảm xúc các bình luận:

+ Xây dựng lớp IdentifyCommentsMapper

* + - * Đầu vào: Cặp key/value biểu diễn dữ liệu các bình luận
      * Xử lý: Đọc từng dòng bình luận, tách mã định danh (ID) và nội dung bình luận (Text) chuyển đổi thành cặp key/value với Text là key và ID là value.
      * Đầu ra: Cặp key/value của bình luận

+ Xây dựng lớp IndentifyCommentsReducer

* + - * Đầu vào: Cặp key/value được chuyển từ hàm Mapper
      * Xử lý:
        + Đọc từ khóa từ file keyword.txt

Kiểm tra sự xuất hiện của từ khóa trong từng bình luận

* + - * + Đếm số lượng từ mang ý nghĩa tích cực và tiêu cực trong bình luận bằng cách so sánh với các file positive.txt và negative.txt
        + Tính giá trị biến ovrall dựa trên số lượng từ tích cực và tiêu cực
        + Viết cảm xúc của bình luận vào file kết quả output.txt
      * Đầu ra: keyOut (Nhận xét và đánh giá tổng quan về từ khóa)/ valOut (giá trị overall của cảm xúc)

# CHƯƠNG 3: ĐỀ XUẤT DỰA VÀO KHAI PHÁ DỮ LIỆU PHÂN CỤM

## Giới thiệu về phân cụm (clustering)

* *Giới thiệu*: Phân cụm dữ liệu là bài toán gom nhóm các đối tượng dữ liệu vào thành từng cụm (cluster) sao cho các đối tượng trong cùng một cụm có sự tương đồng theo một tiêu chí nào đó, là một phương thức khai phá dữ liệu quan trọng. Việc phân tích gom cụm là tìm hiểu giải thuật và phương thức phân loại đối tượng. Một cụm là tập các đối tượng tương tự nhau hoặc là tập các thực thể hoặc nhóm các đối tượng mà các thực thể trong cùng cụm phải giống nhau; các thực thể ở các cụm khác nhau thì không giống nhau. Mỗi thực thể có thể có nhiều thuộc tính hoặc tính năng giống nhau được đo sự giống nhau dựa trên các thuộc tính hoặc tính năng giống nhau.
* *Ý tưởng*: Bằng cách phân cụm dữ liệu đã chọn, đánh giá kết quả để từ đó đưa ra đề xuất của cá nhân sao cho phù hợp với từng cụm dữ liệu.
* *Thuật toán sử dụng*: Kmean clustering

## Giới thiệu thuật toán Kmean clustering:

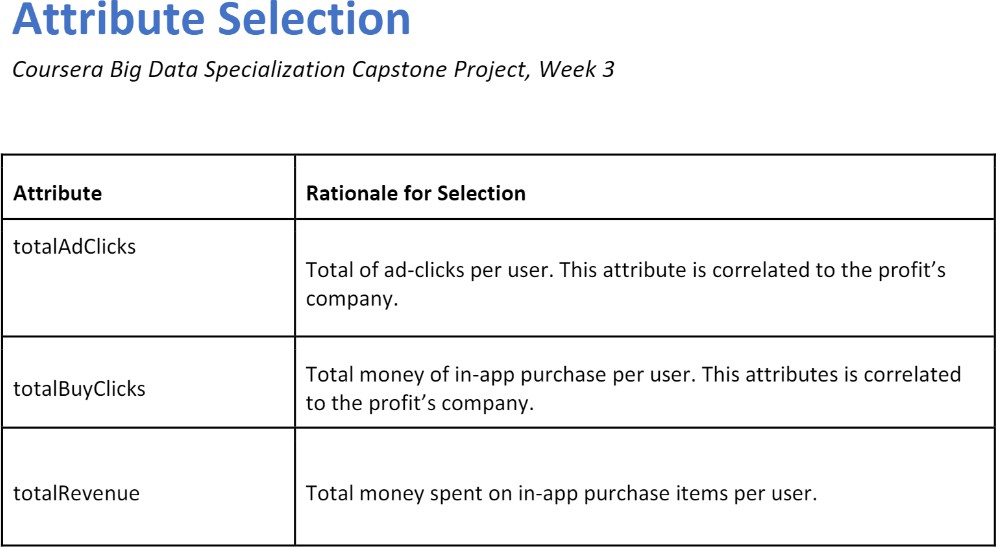
Giải thuật K-Means lấy tham số đầu vào là k và chia một tập n đối tượng thành k cụm dựa vào việc đo độ tương tự. Trị trung bình hay các tâm là tổng độ đo tương tự của các đối tượng dữ liệu trong cùng cụm. Các đối tượng còn lại sẽ được gán cho cụm gần nhất, dựa vào khoảng cách tới các tâm khác nhau, sau đó tính lại các tâm. Quá trình này được lặp lại cho đến khi hàm điều kiện hội tụ

Mô tả:

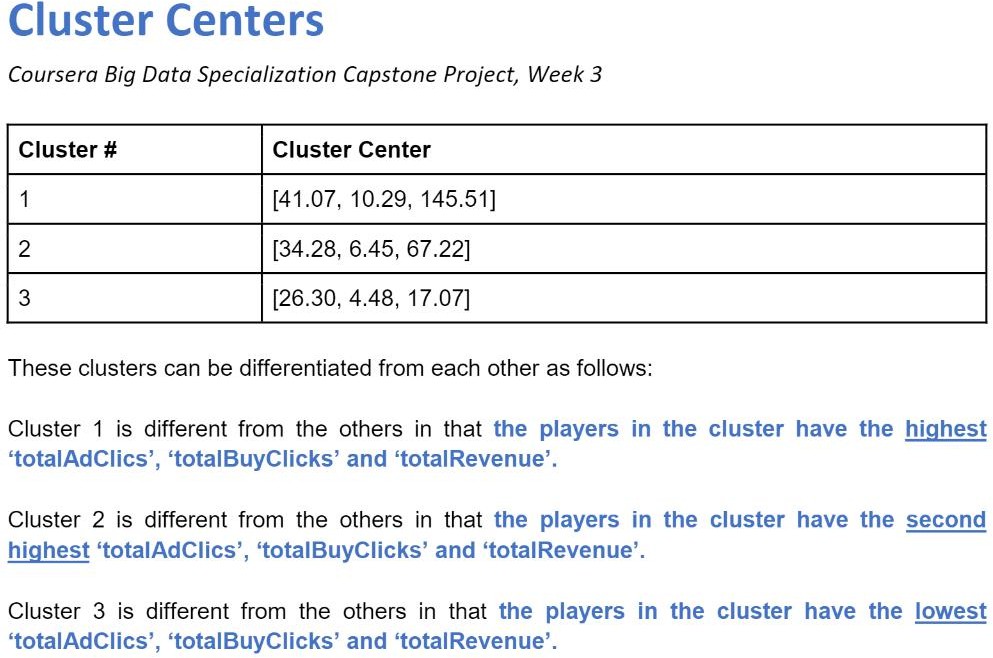
* *Input*: Số cụm k và n tài liệu.
* *Output*: k cụm. Các bước thực hiện:
* Bước 1: Chọn ngẫu nhiên k tài liệu từ n tài liệu để khởi tạo làm tâm các cụm
* Bước 2: Tính khoảng cách từ các tài liệu còn lại đến các tâm của các cụm, gán mỗi tài liệu đó cho cụm gần nhất.
* Bước 3: Tính toán và điều chỉnh tâm các cụm.
* Bước 4: Lặp bước 2 và bước 3 cho đến khi hàm điều kiện hội tụ. Kết thúc chương trình

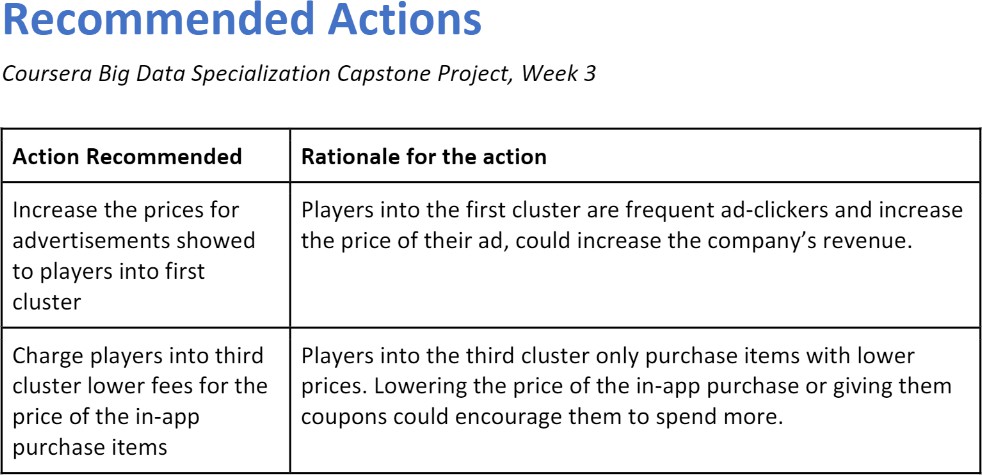
## Triển khai bài toán đề xuất dựa trên phân cụm

## Ví dụ minh họa được đưa ra ở phần hướng dẫn

Bước 1: Sử dụng bộ dữ liệu được cung cấp sẵn, chọn ra một vài thuộc tính (feature) phù hợp.

Bước 2: Sử dụng thuật toán Kmean, tìm ra k tâm cụm (trong ví dụ trên k = 3)

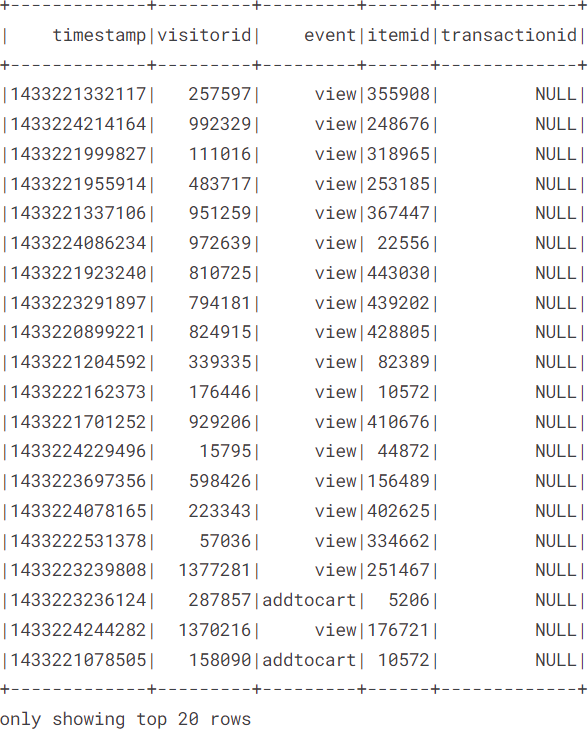


Bước 3: Đưa ra đề xuất của cá nhân, dựa trên đánh giá của bản thân về tâm cụm đã tìm được.

## Mô tả bài toán do nhóm đề ra

*Input:* Bộ dữ liệu bao gồm 3 tệp: dữ liệu hành vi người dùng, thuộc tính mặt hàng, cây danh mục. Dữ liệu được thu thập từ một trang web thương mại điện tử thực tế. *Output:* Phân cụm dữ liệu, tìm ra các thuộc tính của dữ liệu nhằm mục đích đưa ra đề xuất tốt hơn cho doanh nghiệp.

## Các bước triển khai

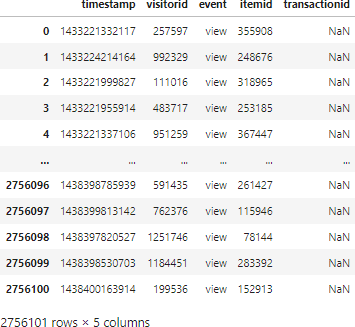
Bước 1: Kiểm tra dữ liệu trong tài liệu đã đề cập. Dữ liệu gồm ~2 triệu dòng đối với dữ liệu hành vi người dùng và 20 triệu dòng đối với dữ liệu thuộc tính mặt hàng

Bước 2: Thực hiện truy vấn để chọn ra từ tập dữ liệu các feature ta muốn sử dụng cho việc phân cụm.

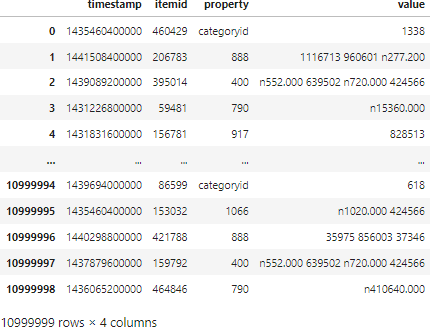
Bước 3: Sử dụng Pyspark cho thuật toán Kmeans, phân cụm dữ liệu.

Bước 4: Tìm ra các tâm cụm, đánh giá, tìm ra và giải quyết vấn đề (nếu có). Từ kết quả thu được, đưa ra đề xuất của cá nhân.

## Demo chương trình cài đặt

* + 1. Kiểm tra thông tin ban đầu của tập dữ liệu và kiểm tra phân vùng của dữ liệu trên PySpark:

*Hình 12. Dữ liệu hành vi người dùng*



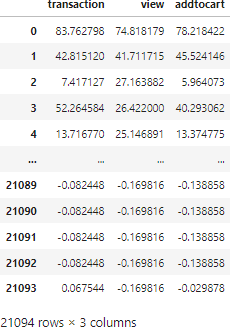
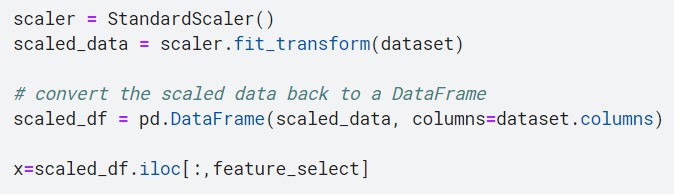
*Hình 13. Dữ liệu thông tin mặt hàng*



* + 1. Thực hiện các truy vấn để tạo ra dữ liệu sử dụng cho thuật toán Kmeans:

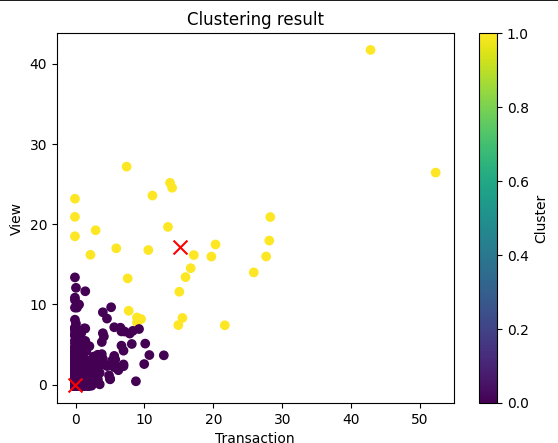


* + 1. Scale lại dữ liệu trước khi sử dụng mô hình học máy:



*Hình 14. Dữ liệu sau khi scale*

* + 1. Sử dụng Pyspark chạy Kmean, dùng metric Silhouette coefficient đánh giá chất lượng cụm (đọc thêm về Silhouette coeficient ở tài liệu tham khảo)
    2. Trực quan hóa dữ liệu cụm



* + 1. Đưa ra đề xuất

Bằng việc phân tập dữ liệu thành 2 cụm, ta đi đến các phân tích như sau:

* + - * Hai cụm sẽ lần lượt đại diện cho hai phân khúc người dùng - những người thường xuyên sử dụng nền tảng thương mại được đề cập, và những người ít sử dụng nền tảng ấy.
      * Cụm những người thường dùng (màu vàng) sẽ có số lượt xem và mua sản phẩm cao, còn cụm những người ít sử dụng trang thương mại (màu tím) sẽ có lượt xem và mua ít.
      * Một điểm đáng chú ý là cụm những người được coi là ít sử dụng có tâm cụm không nằm xấp xỉ chính giữa phân bố các điểm dữ liệu, mà bị kéo lệch về gần điểm 0. Điều này có nghĩa là có rất nhiều người dùng tạo tải khoản trên nền tảng thương mại kể trên, tuy nhiên không sử dụng (dẫn đến lượt xem và mua bằng 0). Có thể coi những điểm dữ liệu như vậy là outlier, tuy nhiên việc phát hiện ra có nhiều tải khoản người dùng không được sử dụng có ảnh hưởng đến việc đưa ra đề xuất cho doanh nghiệp, nên ta không loại các điểm này đi.

Sau khi đưa ra các phân tích như trên, ta đi đến kết luận:

* + - * Đối với những tải khoản outlier, doanh nghiệp có thể đưa ra các sự kiện “người cũ trở vể” nhằm khuyển khích những người dùng đã có tài khoản nhưng không sử dụng tái kích hoạt tải khoản của mình.
      * Đối với những tài khoản ít lượt xem và mua, doanh nghiệp nên tung ra các sự kiện ưu đãi, giảm giá để kích cầu mua sắm.
      * Đối với những tài khoản thường xuyên sử dụng, có thể tận dụng để đưa ra các trò chơi nhận khuyến mại khi mời những người dùng lâu chưa sử dụng tài khoản vào mua sắm cùng, vừa kích cầu, vừa giải quyết vấn đề các “tài khoản bị bỏ hoang”.

# CHƯƠNG 4: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

## Kết luận

Big data đã thách thức đặt ra cho các tổ chức, doanh nghiệp nhiều cơ hội, thách thức và tài sản quý giá. MapReduce chia việc xử lý thành nhiều khối công việc nhỏ, phân tán khắp các nút tính toán, rồi thu thập lại kết quả. Đề tài đã áp dụng mô hình MapReduce để phân tích cảm xúc, phân cụm. Hoàn thành đề tài “**Phân tích cảm xúc từ khóa trong các bình luận & đề xuất dựa vào khai phá dữ liệu phân tích cụm**”, nhóm em đã đạt được một số kết quả như sau:

+ Hiểu tổng quan về Big data, Hadoop, Pyspark, MapReduce.

+ Hiểu chi tiết về bài toán phân tích cảm xúc (sentiment analysis).

+ Hiểu chi tiết về thuật toán Kmeans.

+ Triển khai ý tưởng và giải pháp MapReduce hóa sentiment analysis, kmeans.

+ Xây dựng thành công các chương trình demo cho đề tài.

+ Đánh giá chương trình.

Tuy nhiên kết quả vẫn còn một số hạn chế:

+ Dữ liệu chạy bài toán phân loại cảm xúc vẫn còn ở mức nhỏ.

+ Kmeans vẫn còn cần được cải thiện về khả năng đưa ra đề xuất phù hợp và phức tạp hơn.

## Hướng phát triển

Áp dụng kiến thức về big data, Apache Hadoop, Pyspark, cải tiến và xây dựng ứng dụng phân tích cảm xúc, phân cụm dữ liệu lớn hơn và vào nhiều lĩnh vực khác nhau. Trong quá trình hoàn thành bài tập lớn, nhóm em đã cố gắng tìm hiểu và tham khảo các tài liệu liên quan. Tuy nhiên, thời gian có hạn nên chúng em sẽ không tránh khỏi những thiếu sót, rất mong nhận được sự đóng góp ý kiến của thầy cô và các bạn để báo cáo và kỹ năng của chúng em ngày được hoàn thiện hơn.

Dưới đây là một số đề xuất về phương hướng phát triển trong tương lai:

+ Sử dụng Apache Hive để làm Data Warehousing, cung cấp một giải pháp có thể mở rộng và hiệu quả về chi phí cho việc phân tích và lưu trữ lượng lớn dữ liệu dạng văn bản của chủ đề phân tích cảm xúc.

+ Sử dụng Apache Pig, Apache Hive để làm công cụ xử lý các tập dữ liệu lớn cho quá trình phân tích và tiền xử lý dữ liệu.

+ Sử dụng các phương pháp phức tạp hơn trong chủ đề phân tích cảm xúc như tiền xử lý dữ liệu, trích xuất thông tin bằng các mô hình word2vec, sử dụng kĩ thuật phân loại hoặc phân cụm.

# TÀI LIỆU THAM KHẢO

1. [https://github.com/HxnDev/Hadoop-MapReduce-to-Analyze-Sentiment-of-](https://github.com/HxnDev/Hadoop-MapReduce-to-Analyze-Sentiment-of-Keyword)

[Keyword](https://github.com/HxnDev/Hadoop-MapReduce-to-Analyze-Sentiment-of-Keyword)

1. <https://data-flair.training/blogs/installation-of-hadoop-3-on-ubuntu/>
2. <https://courses.uet.vnu.edu.vn/course/view.php?id=11402>
3. [https://www.researchgate.net/publication/366561503\_A\_Review\_on\_Big\_Data\_S](https://www.researchgate.net/publication/366561503_A_Review_on_Big_Data_Sentiment_Analysis_Techniques) [entiment\_Analysis\_Techniques](https://www.researchgate.net/publication/366561503_A_Review_on_Big_Data_Sentiment_Analysis_Techniques)
4. <https://journalofbigdata.springeropen.com/articles/10.1186/s40537-017-0111-6>
5. [https://github.com/AlessandroCorradini/University-of-California-San-Diego-Big-](https://github.com/AlessandroCorradini/University-of-California-San-Diego-Big-Data-Specialization/tree/master/06%20-%20Big%20Data%20-%20Capstone%20Project/03%20-%20Recommending%20Actions%20from%20Clustering%20Analysis) [Data-Specialization/tree/master/06%20-%20Big%20Data%20-](https://github.com/AlessandroCorradini/University-of-California-San-Diego-Big-Data-Specialization/tree/master/06%20-%20Big%20Data%20-%20Capstone%20Project/03%20-%20Recommending%20Actions%20from%20Clustering%20Analysis)

[%20Capstone%20Project/03%20-](https://github.com/AlessandroCorradini/University-of-California-San-Diego-Big-Data-Specialization/tree/master/06%20-%20Big%20Data%20-%20Capstone%20Project/03%20-%20Recommending%20Actions%20from%20Clustering%20Analysis)

[%20Recommending%20Actions%20from%20Clustering%20Analysis](https://github.com/AlessandroCorradini/University-of-California-San-Diego-Big-Data-Specialization/tree/master/06%20-%20Big%20Data%20-%20Capstone%20Project/03%20-%20Recommending%20Actions%20from%20Clustering%20Analysis)

# NHIỆM VỤ CÁC THÀNH VIÊN

|  |  |
| --- | --- |
| **Họ và tên** | **Công việc** |
| Phạm Thị Kim Huệ | + Tìm hiểu tổng quan về Hadoop/Mapreduce.  + Triển khai ý tưởng Mapreduce hóa bài toán Phân tích cảm xúc.  + Phân tích cảm xúc bằng Hadoop/Mapreduce.  + Cài đặt chương trình demo và đánh giá.  + Làm báo cáo và slide. |
| Vũ Minh Tiến | + Tìm hiểu tổng quan về Pyspark/Kmean.  + Triển khai thuật toán phân cụm dữ liệu và đưa ra đánh giá.  + Phân cụm với Pyspark, phân tích kết quả, đưa ra đề xuất.  + Cài đặt chương trình demo và đánh giá.  + Làm báo cáo và slide. |
| Phạm Quang Vinh | + Tìm hiểu tổng quan về Pyspark/Mapreduce.  + Triển khai ý tưởng, đặc trưng về dữ liệu cần thu thập.  + Thu thập, xử lý, truy vấn dữ liệu, tích hợp SQL với Pyspark.  + Cài đặt chương trình demo và đánh giá.  + Làm báo cáo và slide. |