**A picture containing picture frame, rectangle, frame

Description automatically generated**⎯⎯⎯&⎯⎯⎯

**ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

**KHOA HỆ THỐNG THÔNG TIN**



**BÁO CÁO ĐỒ ÁN**

**MẠNG XÃ HỘI**

**ĐỀ TÀI: PHÂN TÍCH DỮ LIỆU NHỮNG NGÀNH NGHỀ IT ĐƯỢC ĐĂNG TUYỂN TRÊN LINKEDIN**

**Lớp: IS353.O11.HTCL**

**GVHD:** Nguyễn Thị Kim Phụng

**Sinh viên thực hiện:**

Nguyễn Thị Cẩm Vân 20522145

Lưu Thảo Linh 20521532

***Tháng 12/2023, TPHCM***

**ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

**KHOA HỆ THỐNG THÔNG TIN**

⎯⎯⎯&⎯⎯⎯



**BÁO CÁO ĐỒ ÁN**

**MẠNG XÃ HỘI**

**ĐỀ TÀI: PHÂN TÍCH DỮ LIỆU NHỮNG NGÀNH NGHỀ IT ĐƯỢC ĐĂNG TUYỂN TRÊN LINKEDIN**

**Lớp: IS353.O11.HTCL**

**GVHD:** Nguyễn Thị Kim Phụng

**Sinh viên thực hiện:**

Nguyễn Thị Cẩm Vân 20522145

Lưu Thảo Linh 20521532

***Tháng 12/2023, TPHCM***

# LỜI NHẬN XÉT CỦA GIẢNG VIÊN

……………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………….………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………..………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………..………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………..………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………..………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………..………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………..………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………..………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………..……………………………………………………………………………………………………………………………………………………………………………………………………

# LỜI CẢM ƠN

Trên thực tế không có sự thành công nào mà không gắn liền với những sự hỗ trợ, giúp đỡ dù ít hay nhiều, dù trực tiếp hay gián tiếp của người khác. Với lòng biết ơn sâu sắc nhất, đầu tiên em xin gửi lời cảm ơn chân thành đến tập thể **quý Thầy Cô Trường Đại học Công nghệ thông tin – Đại học Quốc gia TP.HCM và quý Thầy Cô khoa Hệ thống thông tin** đã giúp cho nhóm có những kiến thức cơ bản làm nền tảng để thực hiện đề tài này.

Đặc biệt nhóm chúng em xin gửi lời cảm ơn chân thành tới cô **Nguyễn Thị Kim Phụng – giảng viên môn Mạng Xã Hộ**i đã tận tình giúp đỡ, trực tiếp chỉ bảo, hướng dẫn em trong suốt quá trình làm đề tài. Nhờ đó, em đã tiếp thu được nhiều kiến thức bổ ích trong việc vận dụng cũng như kỹ năng làm đề tài. Nếu không có những lời hướng dẫn, dạy bảo của cô thì em nghĩ đề tài này của em rất khó có thể hoàn thiện được. Một lần nữa, em xin chân thành cảm ơn cô.

Cuối cùng, bản thân em đã làm việc hết công suất để hoàn thành tốt đề tài của mình. Xin chân thành cảm ơn!

# MỤC LỤC

[LỜI NHẬN XÉT CỦA GIẢNG VIÊN 3](#_Toc156565700)

[LỜI CẢM ƠN 4](#_Toc156565701)

[MỤC LỤC 5](#_Toc156565702)

[CHƯƠNG 1: TỔNG QUAN 7](#_Toc156565703)

[1.1 Giới thiệu Dataset 7](#_Toc156565704)

[1.2 Xác định bài toán 7](#_Toc156565705)

[1.3 Mô tả dữ liệu 7](#_Toc156565706)

[1.3.1 Nguồn dữ liệu 7](#_Toc156565707)

[1.3.2 Mô tả dữ liệu 7](#_Toc156565708)

[CHƯƠNG II: XỬ LÝ VÀ PHÂN TÍCH DỮ LIỆU 7](#_Toc156565709)

[2.1 Các thư viện được sử dụng 7](#_Toc156565710)

[2.2 Làm sạch dữ liệu 7](#_Toc156565711)

[2.3 Chuyển đổi dataframe thành đồ thị 9](#_Toc156565712)

[2.3.1 Đồ thị 2 phía 9](#_Toc156565713)

[2.3.2 Đồ thị 1 phía 11](#_Toc156565714)

[III. CÁC ĐỘ ĐO VÀ KẾT QUẢ 15](#_Toc156565715)

[3.1 Degree Centrality 15](#_Toc156565716)

[3.1.1 Python 16](#_Toc156565717)

[3.1.2 Gephi 19](#_Toc156565718)

[3.2 Closeness Centrality 19](#_Toc156565719)

[3.2.1 Python 19](#_Toc156565720)

[3.2.2 Gephi 23](#_Toc156565721)

[3.3 Betweenness Centrality 23](#_Toc156565722)

[3.3.1 Python 23](#_Toc156565723)

[3.3.2 Gephi 27](#_Toc156565724)

[3.4 Eigen Vector 27](#_Toc156565725)

[3.4.1 Python 27](#_Toc156565726)

[3.4.2 Gephi 30](#_Toc156565727)

[3.5 Pagerank 30](#_Toc156565728)

[3.5.1 Python 31](#_Toc156565729)

[3.5.2 Gephi 34](#_Toc156565730)

[CHƯƠNG IV: THUẬT TOÁN PHÁT HIỆN CỘNG ĐỒNG 34](#_Toc156565731)

[4.1 Thuật toán Girvan Newman 34](#_Toc156565732)

[4.2 Thuật toán Louvain 38](#_Toc156565733)

[4.3 Thuật toán K-Means 44](#_Toc156565734)

[CHƯƠNG V: ĐỒ THỊ 10 NODE 52](#_Toc156565735)

[5.1 Mô hình Gephi 52](#_Toc156565736)

[5.2 Các độ đo 52](#_Toc156565737)

[5.2.1 Ma trận kề 52](#_Toc156565738)

[5.2.2 Bảng Mapping 52](#_Toc156565739)

[5.2.3 Degree Centrality 52](#_Toc156565740)

[5.2.4 Closeness Centrality 52](#_Toc156565741)

[5.2.5 Betweeness Centrality 52](#_Toc156565742)

[5.2.6 Eigen Vector 52](#_Toc156565743)

[5.2.7 PageRank 52](#_Toc156565744)

[KẾT LUẬN 53](#_Toc156565745)

[TÀI LIỆU THAM KHẢO 53](#_Toc156565746)

# CHƯƠNG 1: TỔNG QUAN

## Giới thiệu Dataset

## Xác định bài toán

* Input: Tập dữ liệu ban đầu trên nguồn dữ liệu Kaggle đã qua tiền xử lý dữ liệu
* Ouput: Đưa ra độ đo, đưa ra cộng đồng phục vụ cho việc phân tích mạng xã hội **“JobLinkined”**

## Mô tả dữ liệu

### 1.3.1 Nguồn dữ liệu

* Link dữ liệu: [LinkedIn\_job\_cleandata | Kaggle](https://www.kaggle.com/datasets/shashankshukla123123/linkedin-job-cleandata/data?fbclid=IwAR0kY497YvSwSFZPvBJF56_u82_aRyOMolzD2R5CTbU88o7rFXwgUpeCmg0)

### 1.3.2 Mô tả dữ liệu

* Mỗi dòng trong tập dữ liệu là một ngành nghề được đăng tải trên LinkedIn
* Dữ liệu gồm 5587 dòng và 15 thuộc tính
* Sau khi xử lý khử trùng còn 403 dòng

|  |  |  |
| --- | --- | --- |
| Tên cột | Kiểu dữ liệu | Ý nghĩa |
| Job\_ID | Int |  |
| Designation | String |  |
| Company\_id | Int |  |
| name | String |  |
| work\_type | String |  |
| involvement | String |  |
| employees\_count | Int |  |
| total\_applicants | Int |  |
| linkedin\_followers | Int |  |
| job\_details | String |  |
| details\_id | Int |  |
| industry | String |  |
| level | String |  |
| City | String |  |
| State | String |  |

# CHƯƠNG II: XỬ LÝ VÀ PHÂN TÍCH DỮ LIỆU

## 2.1 Các thư viện được sử dụng

A screenshot of a computer program

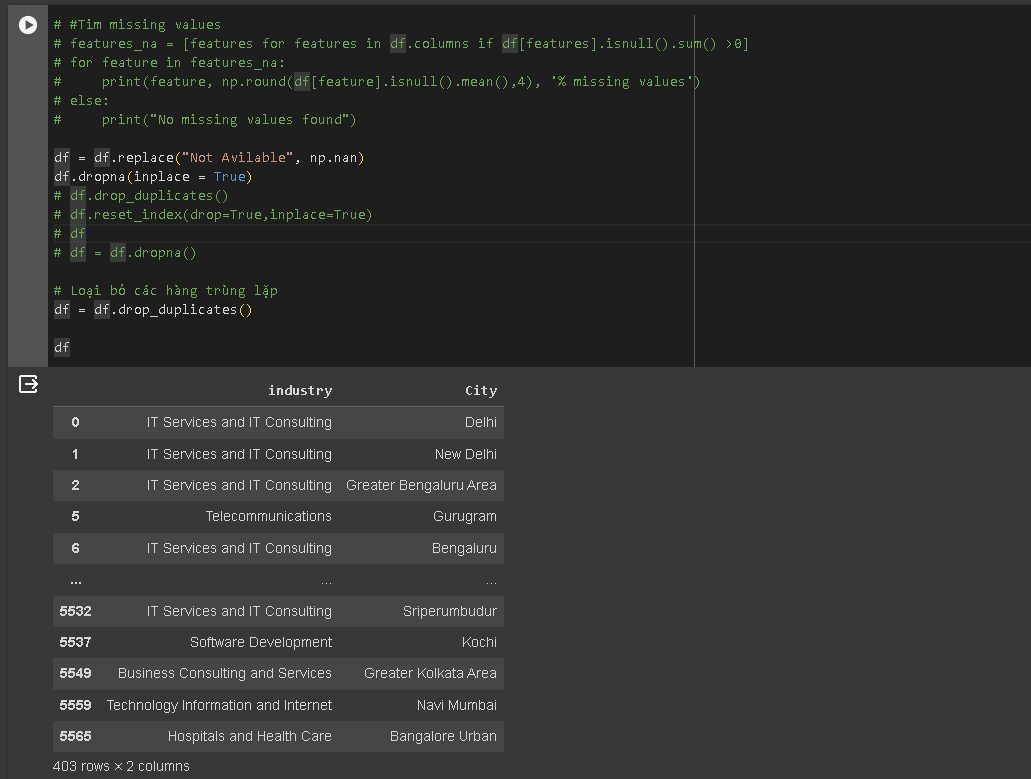
Description automatically generated

## 2.2 Làm sạch dữ liệu

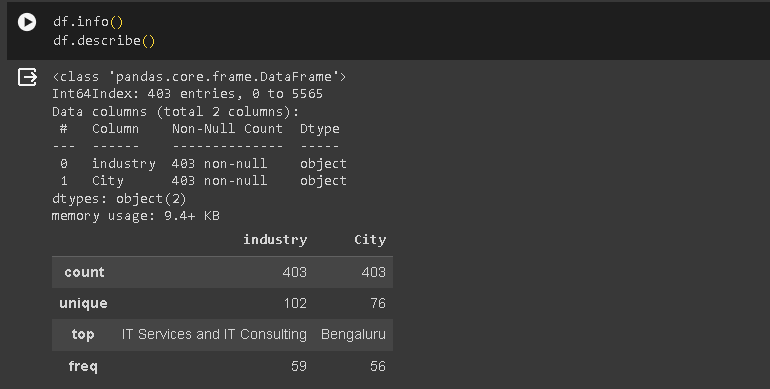
* Đọc dữ liệu từ file CSV và đưa vào dataframe



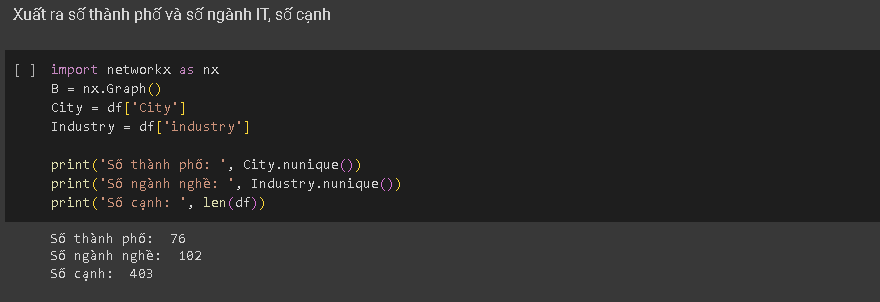
* Làm sạch dữ liệu, xóa dữ liệu trùng lặp



* Sau khi xóa các dữ liệu trống và dữ liệu trùng lặp, Kết quả ta nhận được bộ dữ liệu 403 dòng và 2 cột



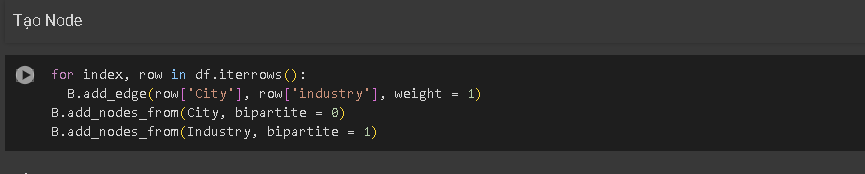
* Kiểm tra số lượng dữ liệu, xuất ra số thành phố, số ngành công nghiệp, số cạnh



## 2.3 Chuyển đổi dataframe thành đồ thị

### 2.3.1 Đồ thị 2 phía

* Tạo node



* **Node: là tên của ngành nghề bà tên các thành phố**
* **Edge: cho ta biết ngành IT được đăng tuyển ở thành phố nào**
* Code hiển thị đồ thị 2 phía

A black screen with white text

Description automatically generated

A screen shot of a computer

Description automatically generated

* Đồ thị 2 phía

A blue and black dotted rectangle

Description automatically generated

**Khi nhìn vào đồ thị 2 phía thì ta có thể thấy 1 thành phố có rất nhiều ngành công nghiệp phát triển và một ngành công nghiệp cũng phát triển ở nhiều thành phố.**

### 2.3.2 Đồ thị 1 phía

* Node: Là các thành phố
* Edge: Hai thành phố có cùng một ngành công nghiệp sẽ hợp thành 1 cạnh
* Code hiển thị đồ thị 1 phía:

A screenshot of a computer program

Description automatically generated

* Đồ thị 1 phía:

A network of cities with red dots

Description automatically generated

Xuất dữ liệu đồ thị một phía ra file csv để thực hiện trên GePhi

Lưu các cạnh của đồ thị vào edges

Tạo dataframe là các cạnh u, v, weight tương đương với source, target, weight

Xuất ra file Gelphi để đưa vào phần mềm Gephi

A screenshot of a computer

Description automatically generated

A screenshot of a cellphone

Description automatically generated

* Xuất ra số đỉnh và số cạnh của đồ thị và kiểm tra đồ thị liên thông

A screenshot of a computer program

Description automatically generated

# CHƯƠNG III. CÁC ĐỘ ĐO VÀ KẾT QUẢ

**Sau khi import file csv dữ liệu ở trên vào Data Laboratory trên Gephi ta thu được các độ đo cần thiết để so sánh kết quả giũa Gephi và Python**

## 3.1 Degree Centrality

Hệ số này sẽ giúp chúng ta đo lường được số lượng của các mối quan hệ trực tiếp của một actor nào đó với các thành viên khác trong mạng lưới. Giá trị của hệ số này chạy từ 0.00 đến 1.00 và khi giá trị càng gần tới 1.00 thì tính trung tâm trực tiếp của actor càng lớn, tức là càng nằm ở vị trí trung tâm của mạng lưới. Công thức tính như sau:

A picture containing clock

Description automatically generated

Trong đó,

k = Tổng số các mối quan hệ trực tiếp của actor

n = Tổng số actor trong mạng lưới

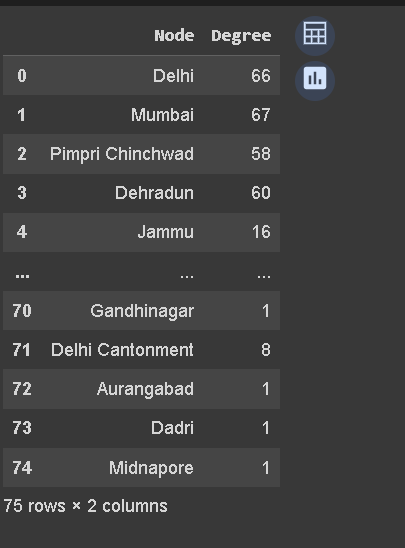
### 3.1.1 Python

* **Code Degree Centrality**

A screen shot of a computer program

Description automatically generated

* **Kết quả**



* **Top 10 Node có Degree Centrality cao nhất**

A screenshot of a computer program

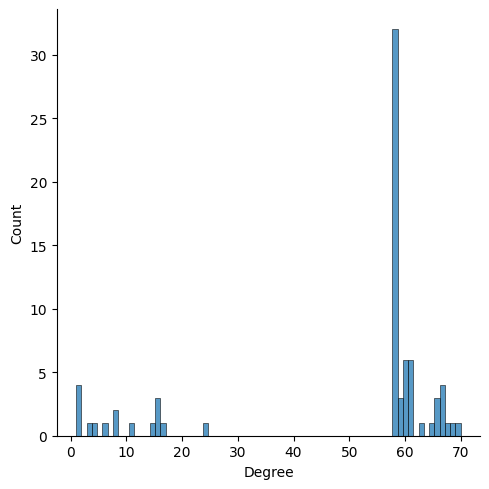
Description automatically generated

* **Top 10 Node có Degree Centrality thấp nhất**

A screenshot of a computer

Description automatically generated

* **Biểu đồ phân bố Degree Centrality**



### 3.1.2 Gephi

A graph with red dots and numbers

Description automatically generated

*Mật độ phân bố Degree Centrality*

* **Top 10 Node có Degree Centrality cao nhất**

**A screenshot of a computer

Description automatically generated**

* **Top 10 Node có Degree Centrality thấp nhất**

A screenshot of a computer

Description automatically generated

## 3.2 Closeness Centrality

Điểm yếu của hệ số trung tâm trực tiếp là nó chỉ tính các mối quan hệ trực tiếp của actor mà thôi nên chưa chắc actor có hệ số trung tâm trực tiếp cao là người "gần gũi" với mọi thành viên khác trong mạng. Tính gần gũi hay lân cận cũng là một trong những tiêu chí quan trọng thể hiện vị thế của actor trong mạng, bởi một actor càng gần gũi với các thành viên trong mạng lưới bao nhiêu thì actor đó càng dễ có nhiều thông tin, càng có nhiều uy thế và do đó càng dễ gây ảnh hưởng lên toàn bộ mạng lưới. Để đo lường hệ số này, chúng ta sẽ tính tổng số "bước" (step) của "đoạn đường" ngắn nhất (geodesic path) mà actor phải "đi" để đến được với tất cả các thành viên khác trong mạng lưới.

Hệ số này cũng có giá trị đi từ 0.00 đến 1.00, càng gần đến 1.00 thì actor càng gần với mọi thành viên khác trong mạng lưới, tức đoạn đường phải đi để đến với mọi actor khác càng ngắn và ngược lại. Công thức tính hệ số này như sau:

Diagram

Description automatically generated with low confidence

Trong đó:

n = Tổng số actor trong mạng lưới

∑d (x,y) = Tổng số "bước" (step) của đoạn đường ngắn nhất mà actori phải đi để đến với mọi actor trong mạng

### 3.2.1 Python

* **Code Closeness Centrality**

A screenshot of a computer program

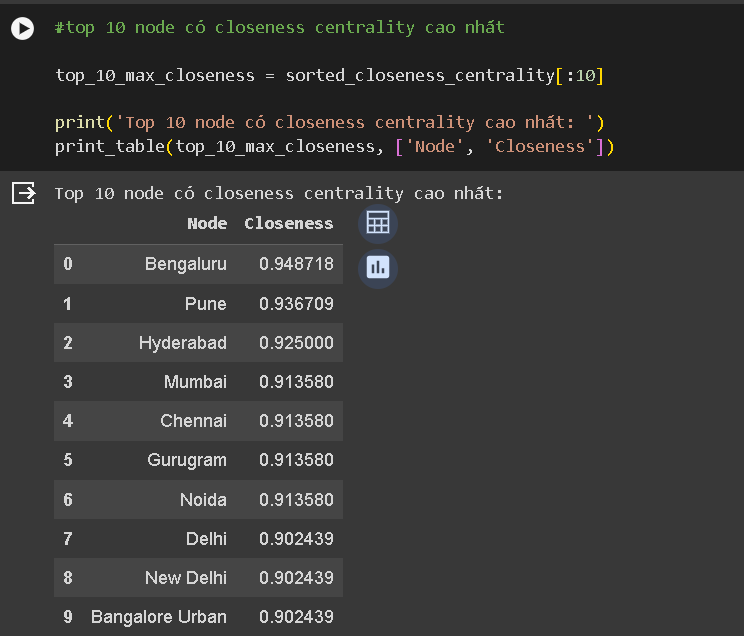
Description automatically generated

* **Kết quả**

A screenshot of a computer

Description automatically generated

* **Top 10 Node có Closeness Centrality cao nhất**



* **Top 10 Node có Closeness Centrality thấp nhất**

A screenshot of a computer

Description automatically generated

* **Biểu đồ phân bố Closeness Centrality**

A graph of a number of objects

Description automatically generated

→ Độ do closeness là độ gần khoảng cách của các node với các node khác trong mạng, độ đo càng cao thì thành phố đó xuất hiện ngành công nghiệp có chung cạnh nhiều nhất với các thành phố khác.

### 3.2.2 Gephi

A graph with red dots

Description automatically generated

* **Top 10 Node có Closeness Centrality cao nhất**

**A screenshot of a computer

Description automatically generated**

* **Top 10 Node có Closeness Centrality thấp nhất**

**A screenshot of a computer

Description automatically generated**

## 3.3 Betweenness Centrality

Theo quan điểm của Freeman, một actor nào đó trong mạng lưới có thể ít gắn kết với các thành viên khác trong mạng lưới (tức hệ số trung tâm trực tiếp thấp), cũng không "gần gũi" lắm với mọi thành viên trong mạng lưới (tức hệ số trung tâm lân cận thấp), nhưng lại là "cầu nối" (bridge), là "nhà trung gian" cần thiết trong mọi cuộc trao đổi trong mạng lưới.

Hệ số này cũng đi từ 0.00 đến 1.00. Khi một actor nào đó có hệ số trung tâm trung gian càng gần đến 1.00 thì số lượng quan hệ giữa các actor khác phải "thông qua" actor này càng nhiều và do đó ảnh hưởng của actor cũng càng lớn.

Cách tính hệ số trung tâm trung gian như sau:

Text

Description automatically generated with low confidence

Trong đó

n(j,z;x) = Tổng số lần làm "trung gian" của actori

n = Tổng số actor trong mạng

### 3.3.1 Python

* **Code Betweenness Centrality**

A screenshot of a computer program

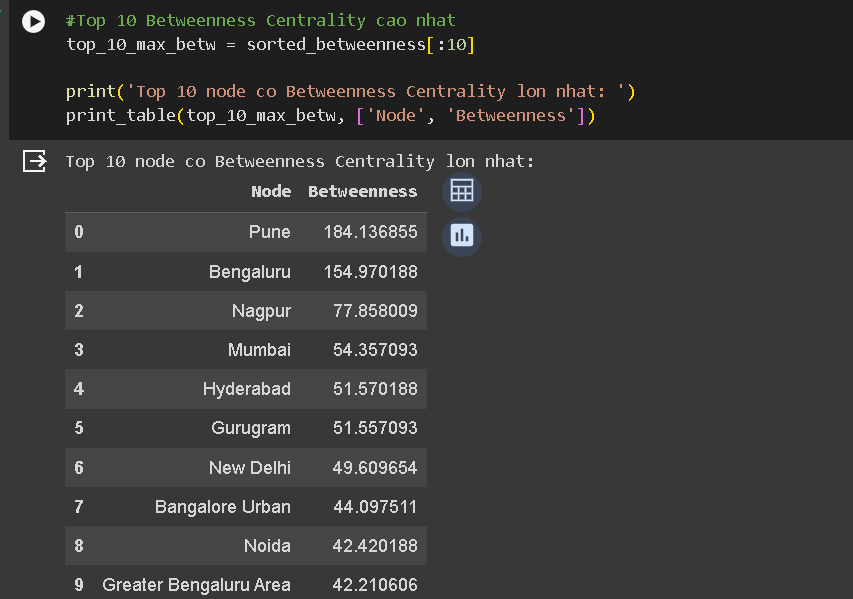
Description automatically generated

* **Kết quả**

A screenshot of a computer

Description automatically generated

* **Top 10 Node có Betweenness Centrality cao nhất**



* **Top 10 Node có Betweenness Centrality thấp nhất**

A screenshot of a computer

Description automatically generated

* **Biểu đồ phân bố Betweenness Centrality**

A graph of a number of objects

Description automatically generated

→ Độ đo betweenness là vai trò trung gian của các mối quan hệ giữa các thành phố trong mạng, độ đo càng cao thì sự giao thoa của các thành phố càng lớn

### 3.3.2 Gephi

A graph with red dots

Description automatically generated

* **Top 10 Node có Betweenness Centrality cao nhất**

**A screenshot of a computer

Description automatically generated**

* **Top 10 Node có Betweenness Centrality thấp nhất**

**A screenshot of a computer

Description automatically generated**

## 3.4 Eigen Vector

Egenvector là các vectơ (khác 0) không thay đổi hướng khi bất kỳ phép biến đổi tuyến tính nào được áp dụng. Nó chỉ thay đổi bởi một hệ số vô hướng. Tóm lại, chúng ta có thể nói, nếu A là một phép biến đổi tuyến tính từ một không gian vectơ V và **x** là một vectơ trong V, không phải là một vectơ 0, thì v là một ký hiệu riêng của A nếu A (X) là một bội số vô hướng của **x** .

Một  Eigenspace  của vector **x** bao gồm một tập hợp tất cả các eigenvector có giá trị riêng tương đương gọi chung với vector zero. Mặc dù vậy, vectơ 0 không phải là một ký hiệu riêng.

Giả sử A là ma trận “n × n” và λ là giá trị riêng của ma trận A, khi đó **x** , một vector khác 0, được gọi là eigenvector nếu nó thỏa mãn biểu thức dưới đây;

A **x**  = λ **x**

**x** là một ký tự riêng của A tương ứng với giá trị riêng, λ.

### 3.4.1 Python

* **Code Eigen Vector**

A screen shot of a computer program

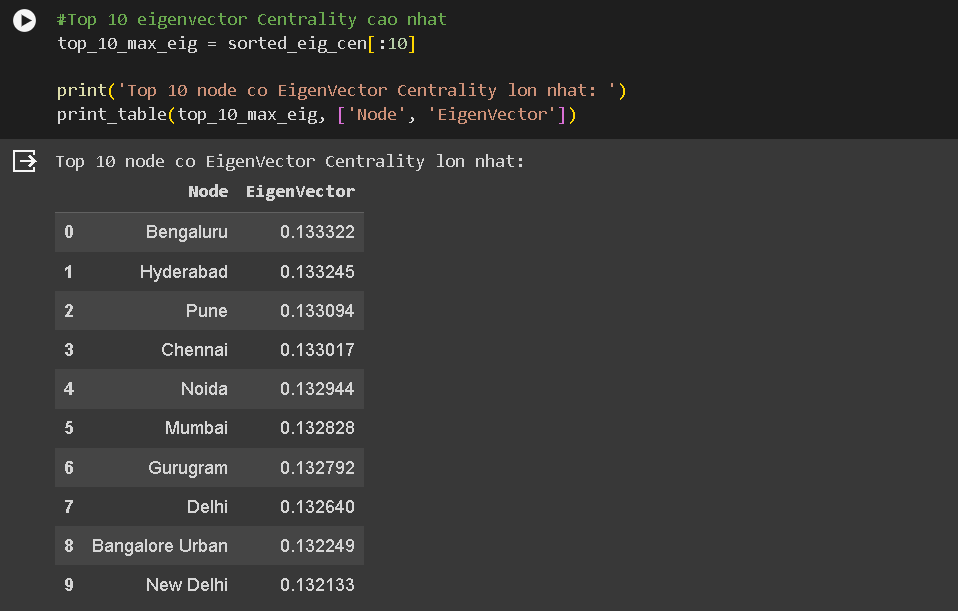
Description automatically generated

* **Kết quả**

A screenshot of a computer

Description automatically generated

* **Top 10 Node có Eigen Vector cao nhất**



* **Top 10 Node có Eigen Vector thấp nhất**

A screenshot of a computer

Description automatically generated

* **Biểu đồ phân bố Eigen Vector**

A screen shot of a graph

Description automatically generated

### 3.4.2 Gephi

A white paper with red dots

Description automatically generated

* **Top 10 Node có Eigen Vector cao nhất**

**A screenshot of a computer

Description automatically generated**

* **Top 10 Node có Eigen Vector thấp nhất**

**A screenshot of a computer

Description automatically generated**

## 3.5 Pagerank

PageRank là thứ hạng trang. PageRank được phát triển tại đại học Stanford bởi Lary Page (cũng bởi vậy mà có tên PageRank) và sau đó bởi Sergey Brin như một phần dự án công cụ tìm kiếm mới.Theo Google một cách tóm lược thì PageRank chỉ được đánh giá từ hệ thống liên kết đường dẫn. Trang của bạn càng nhận nhiều liên kết ( phải là dofollow ) trỏ đến thì mức độ quan trọng trang của bạn càng tăng.

Công thức PageRank có dạng như sau:

A picture containing text, watch, clock, gauge

Description automatically generated

Trong đó:

d: hằng số Google quy định. Thông thường, d = 0.85.

PR(pj): PageRank của các đỉnh đi vào đỉnh i.

L(j): số link out của các đỉnh đi vào đỉnh i.

### 3.5.1 Python

* **Code PageRank**

A screen shot of a computer code

Description automatically generated

* **Kết quả**

A screenshot of a computer

Description automatically generated

* **Top 10 Node có PageRank cao nhất**

A screenshot of a computer program

Description automatically generated

* **Top 10 Node có PageRank thấp nhất**

A screenshot of a computer program

Description automatically generated

* **Biểu đồ phân bố PageRank**

A screenshot of a computer

Description automatically generated

### 3.5.2 Gephi

A white sheet with black lines

Description automatically generated with medium confidence

* **Top 10 Node có PageRank cao nhất**

**A screenshot of a computer

Description automatically generated**

* **Top 10 Node có PageRank thấp nhất**

**A screenshot of a computer

Description automatically generated**

# CHƯƠNG IV: THUẬT TOÁN PHÁT HIỆN CỘNG ĐỒNG

## 4.1 Thuật toán Girvan Newman

Thuật toán lần đầu tiên được đề xuất bởi Freeman. Theo Freeman, các cạnh được coi là cạnh có số lượng con đường ngắn nhất giữa các cặp đỉnh khác nhau chạy qua nó. Cạnh nối có ảnh hưởng rất lớn đến dòng chảy của thông tin giữa các nút khác, đặc biệt là trong trường hợp thông tin lưu truyền trong mạng chủ yếu theo con đường ngắn nhất. Thuật toán điển hình nhất trong các thuật toán chia này là thuật toán Girvan-Newman. Để tìm các cạnh trong mạng nối hai đỉnh thuộc hai cộng đồng khác nhau, khái quát đây là cạnh có độ trung gian cao, và xác định độ do trung gian này bằng cách tính số đường đi ngắn nhất giữa các cặp đỉnh mà có qua nó. Với một đồ thị m cạnh và n 13 đỉnh thì thời gian tính toán cho giai đoạn này là O(mn) .Với đồ thị có trọng số, độ đo trung gian của cạnh có trọng số đơn giản được tính bằng độ đo trung gian của cạnh không có trọng số chia cho trọng số của cạnh đó. Nếu một mạng lưới bao gồm các cộng đồng hoặc nhóm chúng chỉ được liên kết nối yếu bằng một nhóm cạnh, thì tất cả các đường đi ngắn nhất giữa các cộng đồng khác nhau sẽ phải đi dọc theo một trong số ít các cạnh thuộc nhóm cạnh đó. Vì vậy, các cạnh kết nối các cộng đồng sẽ là cạnh có độ đô trung gian cao. Bằng cách loại bỏ các cạnh, thuật toán Girvan-Newman tách được thành các nhóm riêng biệt.

Thuật toán được thực hiện theo các bước sau:

1. Tính độ đo trung gian cho tất cả các cạnh trong mạng.

2. Hủy bỏ các cạnh có độ trung gian cao nhất.

3. Tính lại độ trung gian cho tất cả các cạnh bị ảnh hưởng theo các cạnh đã loại bỏ.

4. Lặp lại từ bước 2 cho đến khi không còn các cạnh trung gian.

* **Code cài đặt và sử dụng thuật toán:**

A screen shot of a computer program

Description automatically generated

* **Đồ thị phân cụm sử dụng Girvan Newman**

A network of cities with blue dots

Description automatically generated with medium confidence

* **In ra số lượng Cluster**

A screen shot of a computer program

Description automatically generated

A screen shot of a computer

Description automatically generated

* **Danh sách các cụm: 2 cụm**

**Cụm 1**: Pimpri Chinchwad | Alipur | Gurugram | Bengaluru East | Dadri | Thane | Chandigarh | Lucknow | Jammu | Agra | Mohali district | Greater Delhi Area | Greater Hyderabad Area | Visakhapatnam | Ghaziabad | Vishakhapatnam | Jalandhar | Greater Kolkata Area | Bengaluru | New Delhi | Ludhiana | Dehradun | Mumbai Metropolitan Region | Ernakulam | Navi Mumbai | Gurgaon | Delhi Cantonment' | Greater Ahmedabad Area | Kolkata | Bangalore Urban | Pune/Pimpri-Chinchwad Area | Vijayawada | Gautam Buddha Nagar | Silao | Aurangabad | Kozhikode | Jamshedpur | Faridabad | Srinagar | Ahmednagar | Mangaluru | Bhubaneswar | Bhubaneshwar | Patna | Mumbai | Midnapore | Greater Bengaluru Area | Noida | Bhopal | Greater Chennai Area | Thiruvananthapuram | Hyderabad | Sriperumbudur | Bangalore Urban district | Kanpur | Coimbatore | Chennai | Trivandrum | Amritsar | Greater Coimbatore Area | Nagpur | Surat | Kochi | Kalyan | Indore | Sahibzada Ajit Singh Nagar | Delhi | Ahmedabad | Vadodara | Pune | Rajkot | Raipur | Jaipur | Greater Nagpur Area |

**Cụm** 2: | Gandhinagar |

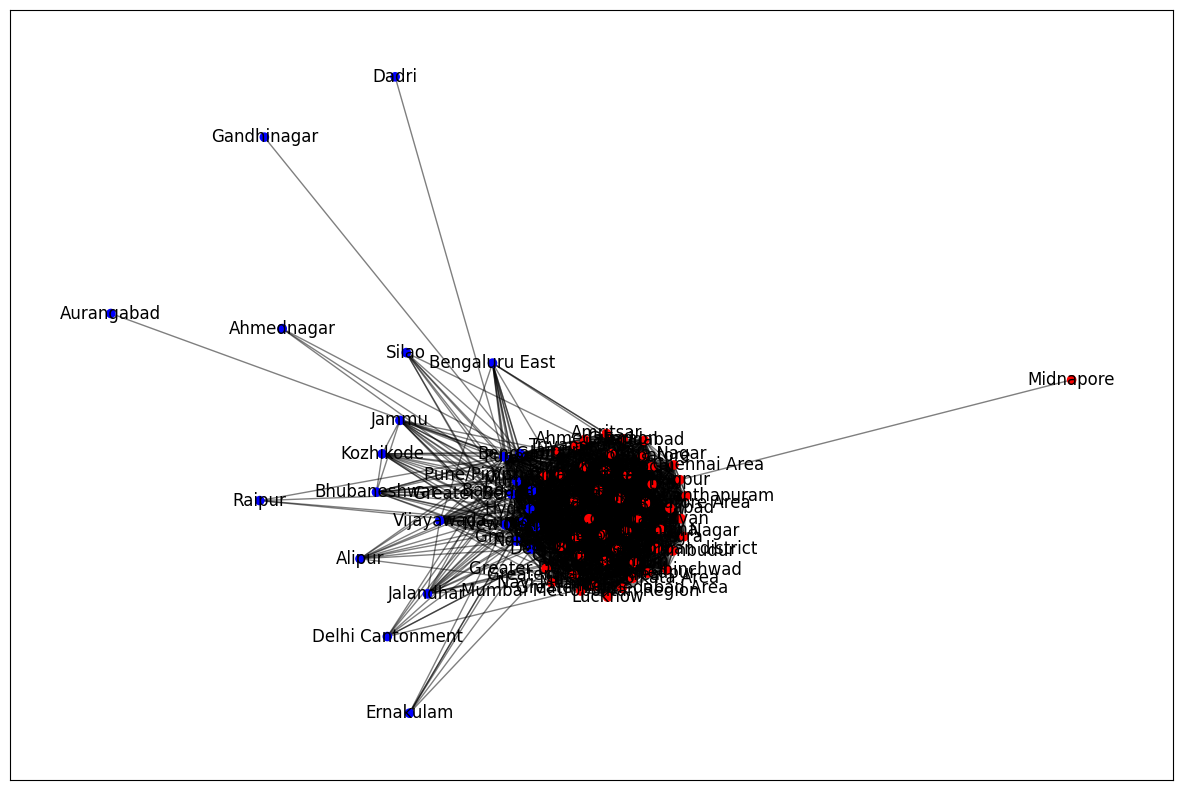
## 4.2 Thuật toán Louvain

Phương pháp Louvain để phát hiện cộng đồng là một thuật toán để phát hiện các cộng đồng trong mạng. Nó tối đa hóa điểm mô-đun cho mỗi cộng đồng, trong đó mô-đun định lượng chất lượng của việc gán các nút cho cộng đồng. Điều này có nghĩa là đánh giá mức độ kết nối của các nút trong cộng đồng với mật độ cao hơn như thế nào so với mức độ kết nối của chúng trong một mạng ngẫu nhiên. Thuật toán Louvain là một thuật toán phân cụm phân cấp, hợp nhất một cách đệ quy các cộng đồng thành một nút duy nhất và thực hiện phân cụm mô-đun trên các đồ thị cô đọng

* **Code cài đặt và sử dụng thuật toán:**



* **Đồ thị phân cụm sử dụng Louvain**

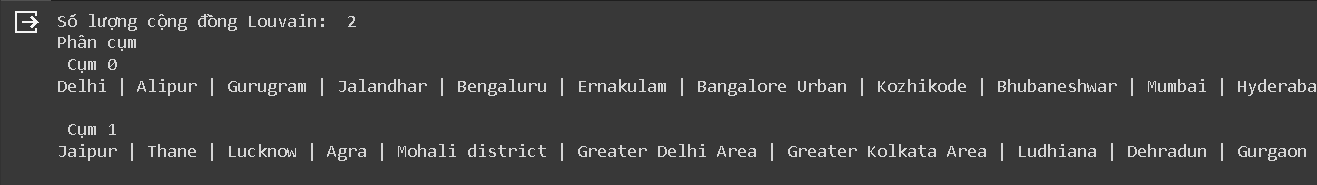


* **In ra số lượng cụm**

A screen shot of a computer program

Description automatically generated

* **Danh sách các cụm: 2 cụm**



**Cụm 1**: Delhi | Alipur | Gurugram | Jalandhar | Bengaluru | Ernakulam | Bangalore Urban | Kozhikode | Bhubaneshwar | Mumbai | Hyderabad | Chennai | Pune | Vijayawada | Bengaluru East | New Delhi | Greater Bengaluru Area | Noida | Jammu | Silao | Raipur | Ahmednagar | Gandhinagar | Delhi Cantonment | Dadri | Aurangabad |

**Cụm 2**: Jaipur | Thane | Lucknow | Agra | Mohali district | Greater Delhi Area | Greater Kolkata Area | Ludhiana | Dehradun | Gurgaon | Greater Ahmedabad Area | Pune/Pimpri-Chinchwad Area | Gautam Buddha Nagar | Jamshedpur | Mangaluru | Bhubaneswar | Greater Chennai Area | Amritsar | Greater Nagpur Area | Surat | Kochi | Sahibzada Ajit Singh Nagar | Ahmedabad | Vadodara | Thiruvananthapuram | Pimpri Chinchwad | Chandigarh | Greater Hyderabad Area | Visakhapatnam | Ghaziabad | Vishakhapatnam | Mumbai Metropolitan Region | Navi Mumbai | Kolkata | Faridabad | Srinagar | Patna | Bhopal | Sriperumbudur | Bangalore Urban district | Kanpur | Trivandrum | Greater Coimbatore Area | Nagpur | Kalyan | Indore | Rajkot | Coimbatore | Midnapore |

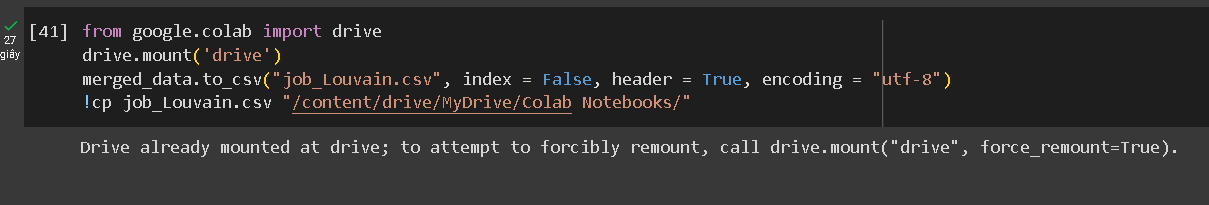
* **Xuất file Excel**

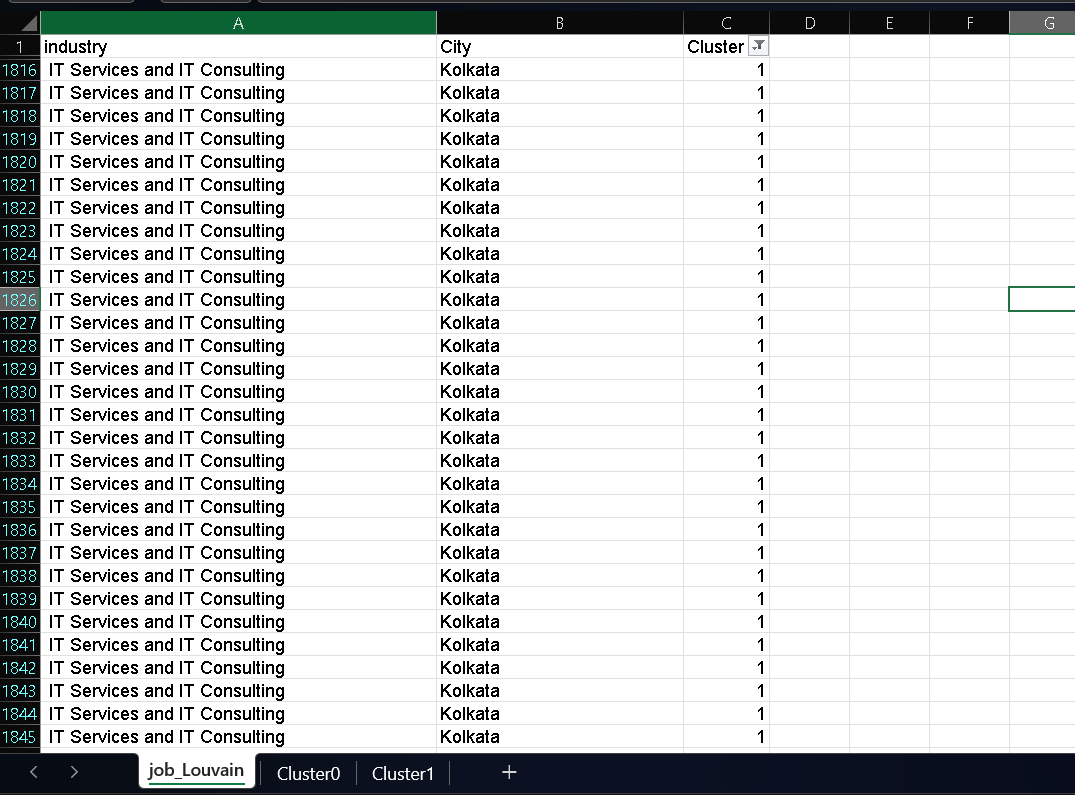
A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated





A screenshot of a computer

Description automatically generated

→ **Ta thấy ở Cluster 0:** Ngành nghề IT Services and IT Consulting được đăng tuyển nhiều nhất ở thành phố Bengaluru

A screenshot of a computer

Description automatically generated

→ **Ta thấy ở Cluster 1:** Ngành nghề IT Services and IT Consulting được đăng tuyển nhiều nhất ở thành phố Kanpur

## 4.3 Thuật toán K-Means

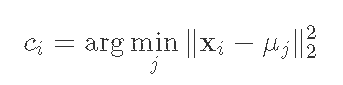
Trong thuật toán k-Means mỗi cụm dữ liệu được đặc trưng bởi một tâm (centroid). tâm là điểm đại diện nhất cho một cụm và có giá trị bằng trung bình của toàn bộ các quan sát nằm trong cụm. Chúng ta sẽ dựa vào khoảng cách từ mỗi quan sát tới các tâm để xác định nhãn cho chúng trùng thuộc về tâm gần nhất. Ban đầu thuật toán sẽ khởi tạo ngẫu nhiên một số lượng xác định trước tâm cụm. Sau đó tiến hành xác định nhãn cho từng điểm dữ liệu và tiếp tục cập nhật lại tâm cụm. Thuật toán sẽ dừng cho tới khi toàn bộ các điểm dữ liệu được phân về đúng cụm hoặc số lượt cập nhật tâm chạm ngưỡng.

Cụ thể các bước của thuật toán k-Means được tóm tắt như sau:

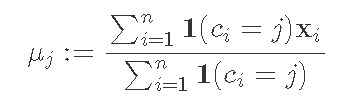
+ Khởi tạo ngẫu nhiên k tâm cụm μ1, μ2, …, μk .

+ Lặp lại quá trình cập nhật tâm cụm cho tới khi dừng:

a. Xác định nhãn cho từng điểm dữ liệu ci dựa vào khoảng cách tới từng tâm cụm:



b. Tính toán lại tâm cho từng cụm theo trung bình của toàn bộ các điểm dữ liệu trong một cụm:



Trong công thức 2.a thì ‖x‖22 là bình phương của norm chuẩn bậc 2, kí hiệu là L2, norm chuẩn bậc 2 là một độ đo khoảng cách thường được sử dụng trong machine learning.

Trong công thức 2.b chúng ta sử dụng hàm 1(.), hàm này có giá trị trả về là 1 nếu nhãn của điểm dữ liệu ci được dự báo thuộc về cụm j, trái lại thì trả về giá trị 0. Như vậy tử số của vế phải trong công thức 2.b chính là tổng khoảng cách của toàn bộ các điểm dữ liệu nằm trong cụm j trong khi mẫu số chính là số lượng các điểm dữ liệu thuộc cụm j. μj chính là vị trí của tâm cụm j mà ta dự báo tại thời điểm hiện tại. Trong thuật toán trên thì tham số mà chúng ta cần lựa chọn chính là số lượng cụm k. Thời điểm ban đầu ta sẽ khởi tạo k điểm dữ liệu một cách ngẫu nhiên và sau đó gán các tâm bằng giá trị của k điểm dữ liệu này. Các bước trong vòng lặp ở bước 2 thực chất là:

a. Gán nhãn cho mỗi điểm dữ liệu bằng với nhãn của tâm cụm gần nhất.

b. Dịch chuyển dần dần tâm cụm μj tới trung bình của những điểm dữ liệu mà được phân về j.

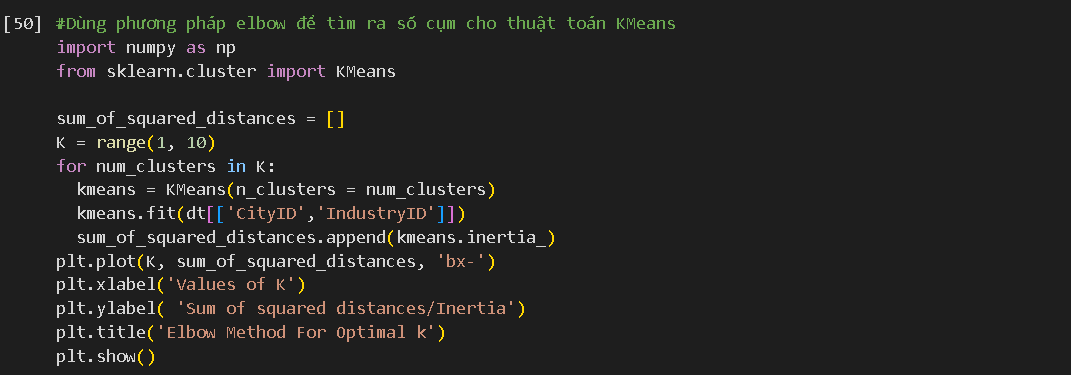
* **Code cài đặt và sử dụng thuật toán:**

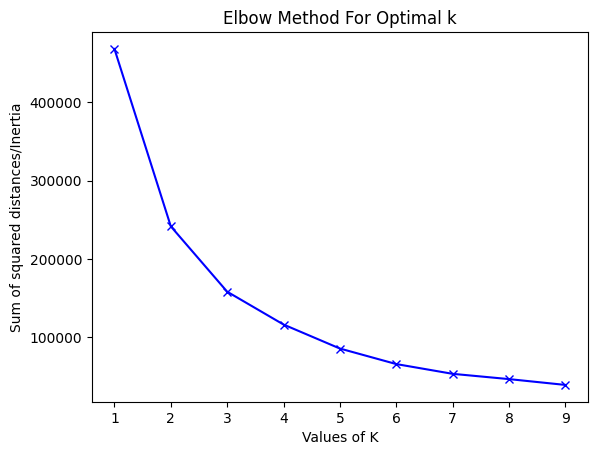
Chuyển đổi cột City và Industry thành encode



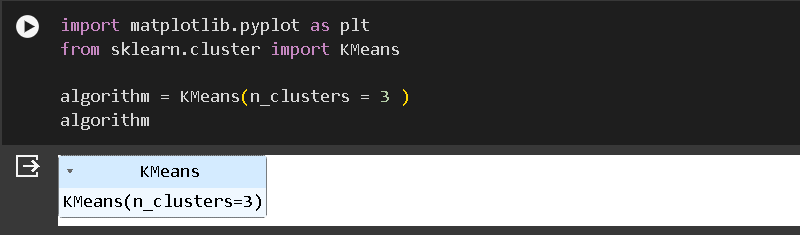


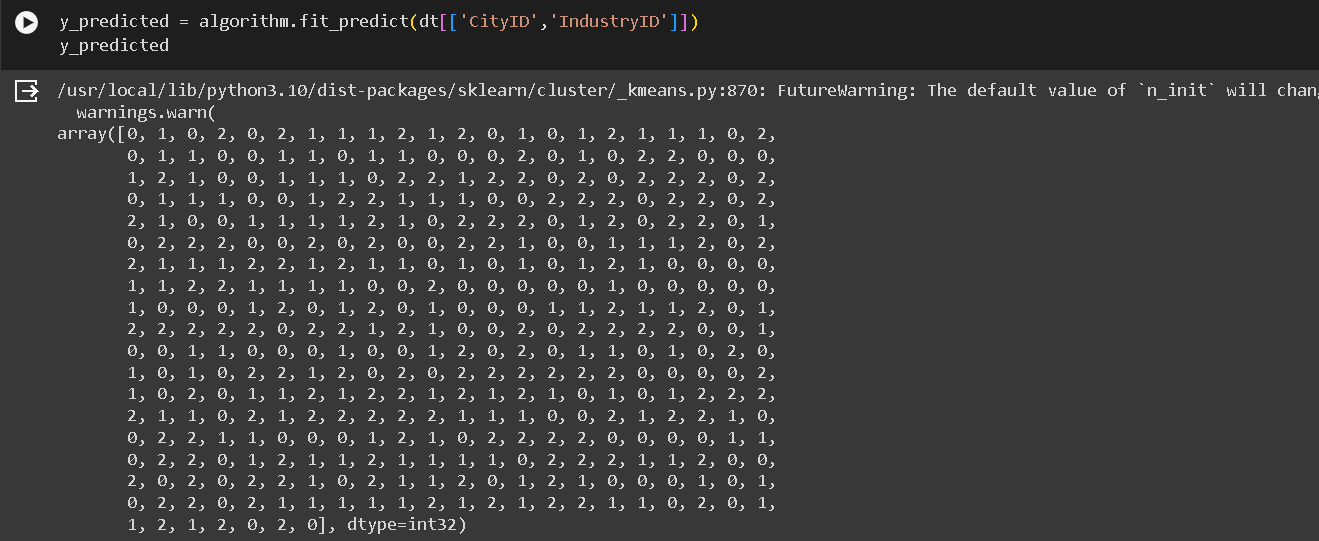
* **Sử dụng phương pháp elbow để tìm số cụm cho thuật toán Kmeans**





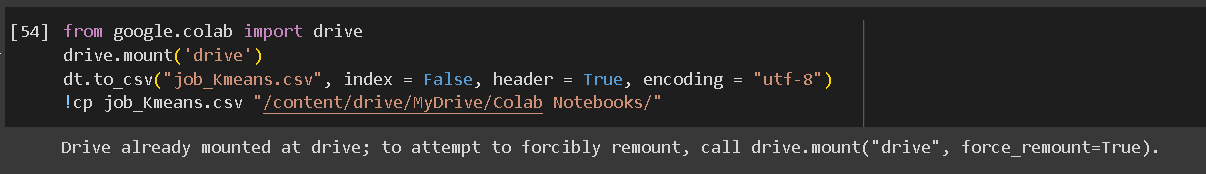
* **Tìm ra số cụm là 3**

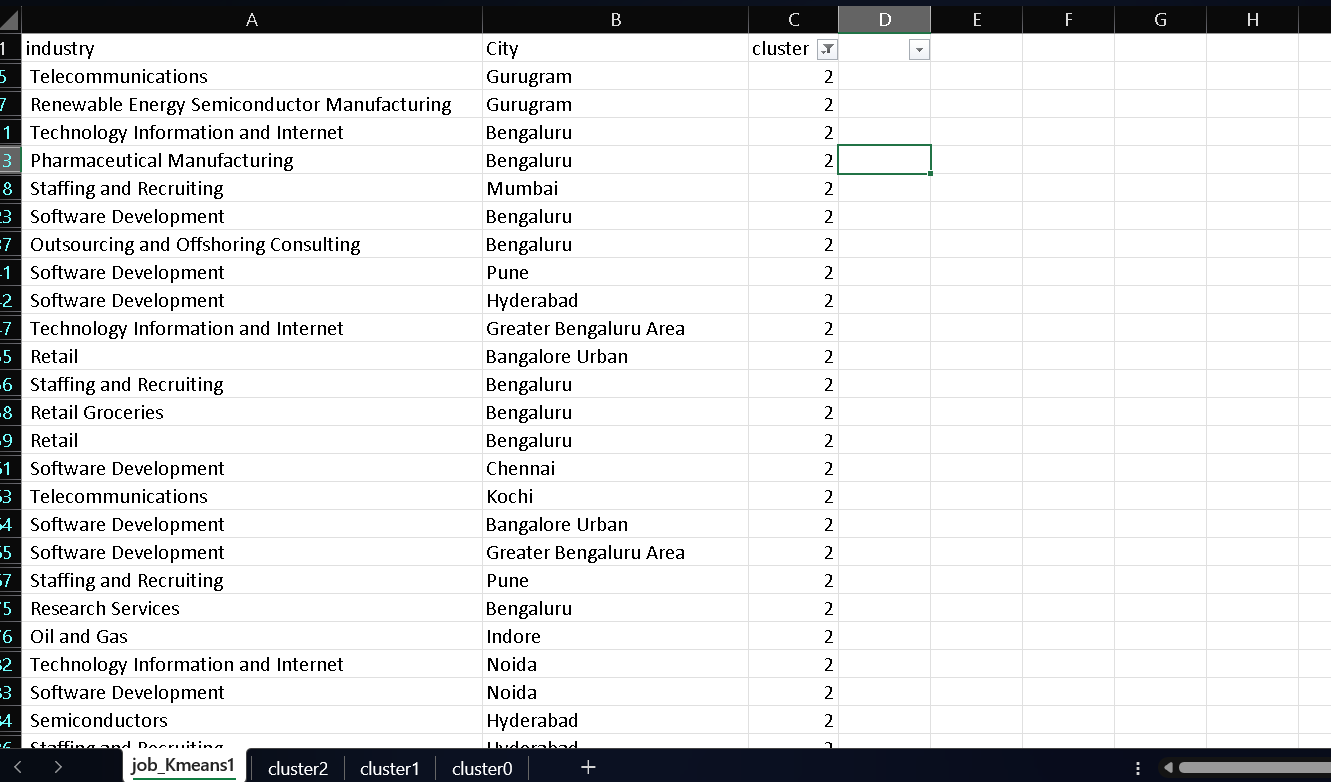




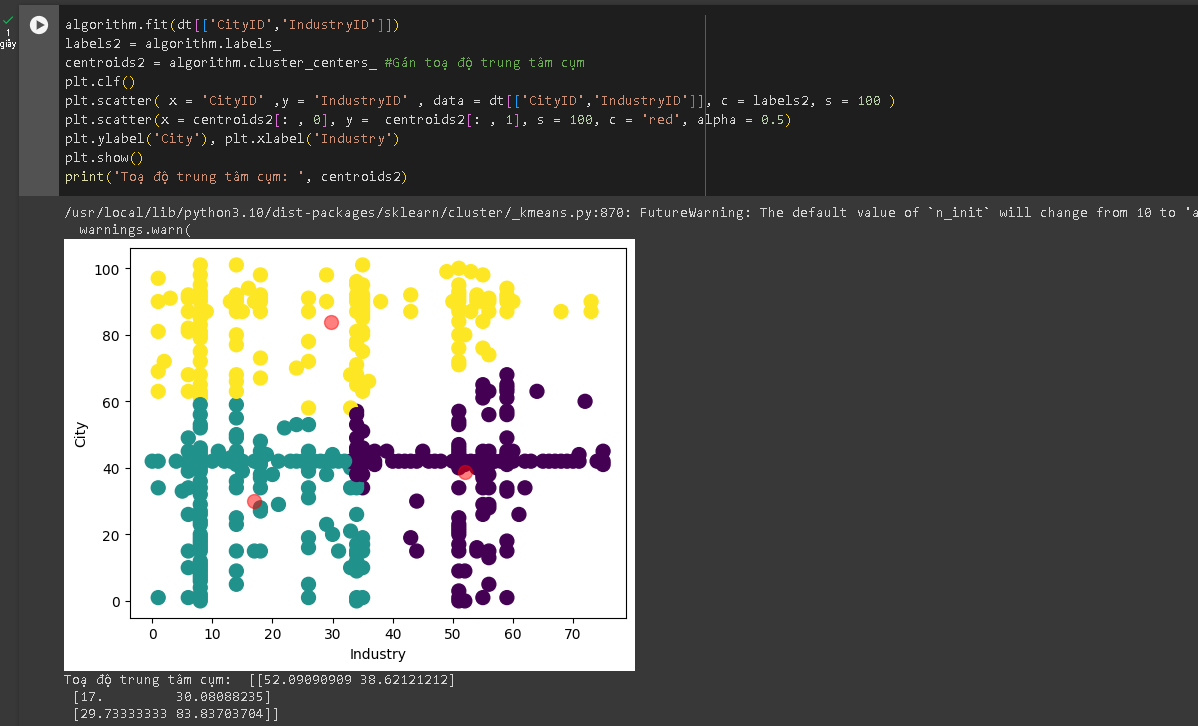
* **Xuất file Excel**

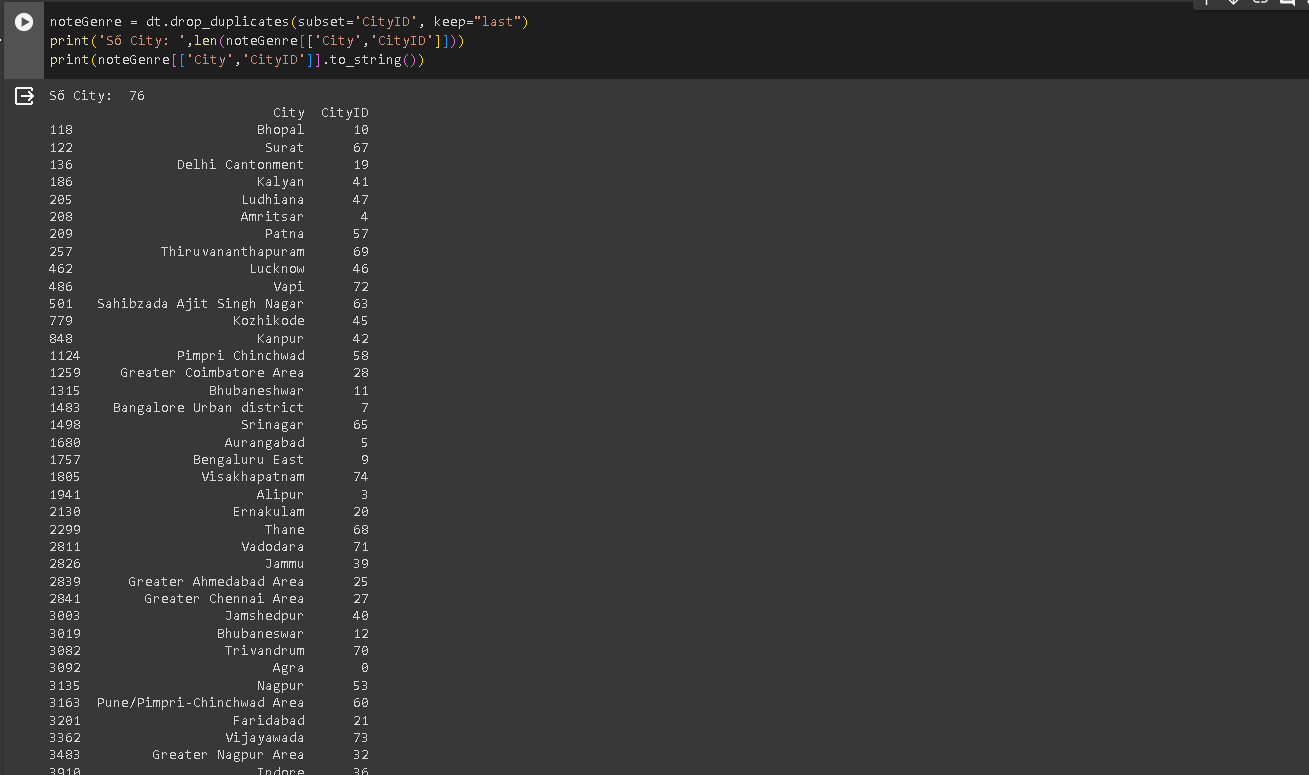
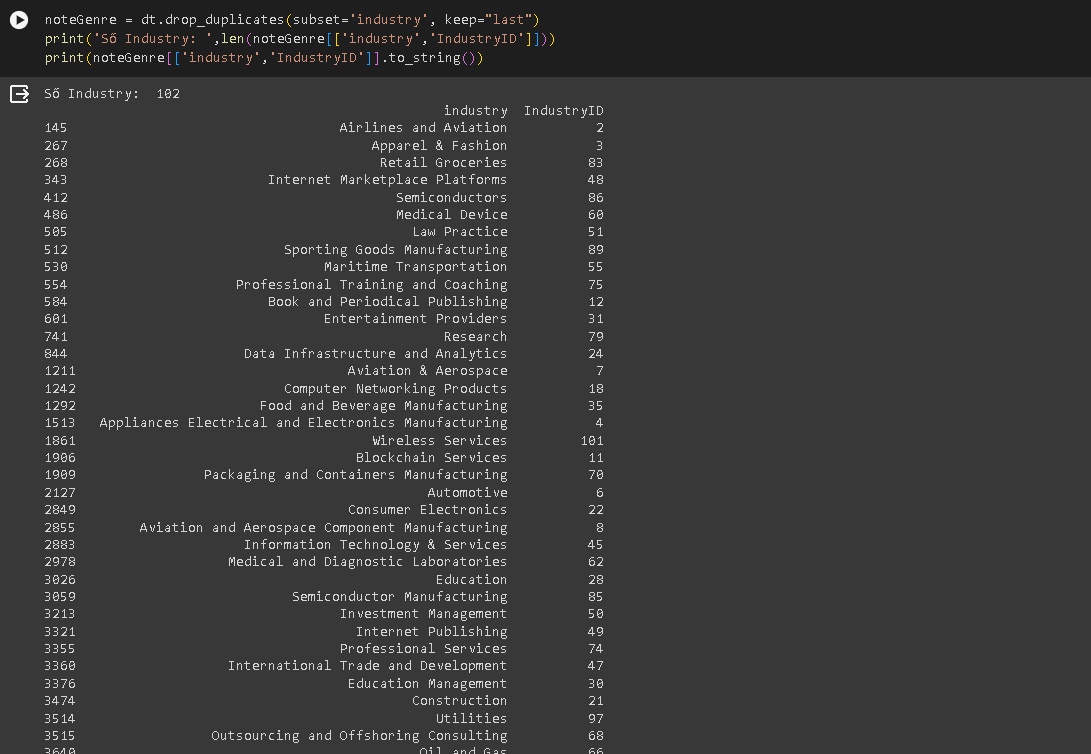






* **Tọa độ trung tâm cụm**



A screenshot of a computer

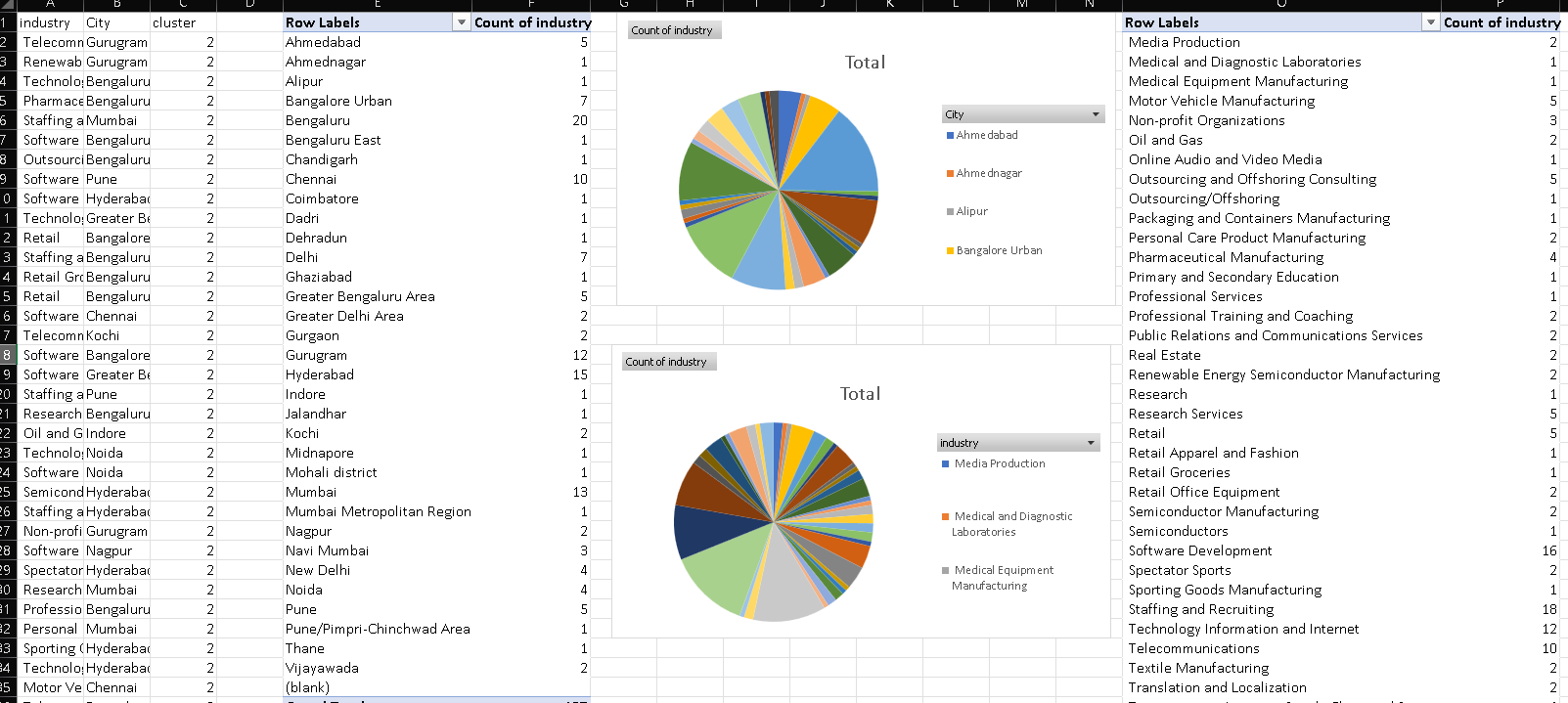
Description automatically generated

→ **Ta thấy ở Cluster 0:** Ngành nghề IT Services and IT Consulting được đăng tuyển nhiều nhất ở thành phố Mumbai

A screenshot of a computer

Description automatically generated

→ **Ta thấy ở Cluster 1:** Ngành nghề IT Services and IT Consulting được đăng tuyển nhiều nhất ở thành phố Bengaluru



→ **Ta thấy ở Cluster 2:** Ngành nghề IT Services and IT Consulting được đăng tuyển nhiều nhất ở thành phố Bengaluru

**Kết luận: Qua 3 thuật toán phát hiện cộng đồng chúng ta nhận ra rằng thuật toán K-Means sử dụng hiệu quả nhất vì thuật toán đã chia nhỏ ra từng cụm và chỉ rõ chi tiết đặc trưng của dataset hơn so với thuật toán Louvain.**

# CHƯƠNG V: TRÍCH XUẤT 10 NODE VÀ THỰC HIỆN TÍNH TAY

Thực hiện trong file Manual Calculation đã nộp trên gg drive

# KẾT LUẬN

# TÀI LIỆU THAM KHẢO