MỤC LỤC

1. **Mô tả dữ liệu**
2. **Tiền xử lý dữ liệu**
3. **Tách và lọc dữ liệu**
4. **Sử dụng K-Mean để thử gom cụm và tìm số cụm có khả năng trong tập dữ liệu**
5. **Kết quả và nhận xét**

# DỮ LIỆU ĐIỂM ĐI THPT QUỐC GIA 2020

1. **Mô tả dữ liệu**

Dữ liệu chứa điểm thi của các thí sinh trên toàn cả nước dự thi kỳ thi Trung học phổ

thông quốc gia (THPTQG) năm 2020, cụ thể gồm:

1. Số thứ tự (chưa đặt tên);

2. Số báo danh (Code);

3. Điểm môn thi Địa lí (Diali);

4. Điểm môn thi GDCD (GDCD);

5. Điểm môn thi Hóa học (Hoahoc);

6. Điểm tổ hợp khối thi Khoa học tự nhiên (KHTN);

7. Điểm tổ hợp khối thi Khoa học xã hội (KHXH);

8. Điểm môn thi Lịch sử (LichSu);

9. Điểm môn thi Ngoại ngữ (Ngoaingu);

10. Điểm môn thi Ngữ Văn (Nguvan);

11.Điểm môn thi Sinh học (Sinhhoc);

12. Điểm môn thi Toán (Toan);

13. Điểm môn thi Vật lí (Vatli);

14. Mã tỉnh/ thành phố của thí sinh dự thi (city).

Căn cứ Điều 3 Chương I — Thông tư số 15/2020/TT-BGDĐT ngày 26/05/2020 của Bộ Giáo dục và đào tạo (tham khảo), Kỳ thi THPTQG “Tổ chức thi 05 bài thi, gồm: 03 bài thi độc lập là Toán, Ngữ văn, Ngoại ngữ; 01 bài thi tổ hợp Khoa học Tự nhiên (viết tắt là KHTN) gồm các môn thi thành phần Vật lí, Hóa học, Sinh học; 01 bài thi tổ hợp Khoa học Xã hội (Viết tắt là KHXH) gồm các môn thi thành phần Lịch sử, Địa lí, Giáo dục công dân đối với thí sinh học chương trình giáo dục phổ thông cấp THPT hoặc các môn thi thành phần Lịch sử, Địa lí đối với thí sinh học chương trình GDTX cấp THPT”. Một số trường hợp đặc biệt có trong dữ liệu có thể xảy ra như sau:

1. Trường hợp thí sinh là thí sinh tự do: chỉ thi những môn thuộc khối thi mà thí sinh lựa chọn để xét tuyển.

2. Thí sinh là thí sinh học chương trình GDTX cấp THPT, trường hợp tham gia thi tổ

hợp Khoa học xã hội chỉ thi Lịch sử và Địa lý.

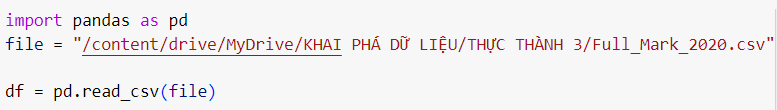
3. Thí sinh đăng ký thi cả 02 bài tô hợp KHTN và KHXH.

Tham khảo trang web https://diemthi.24h.com.vn/, thông tin về tỉnh/ thành phố của thí sinh dự thi được cụ thể hóa:

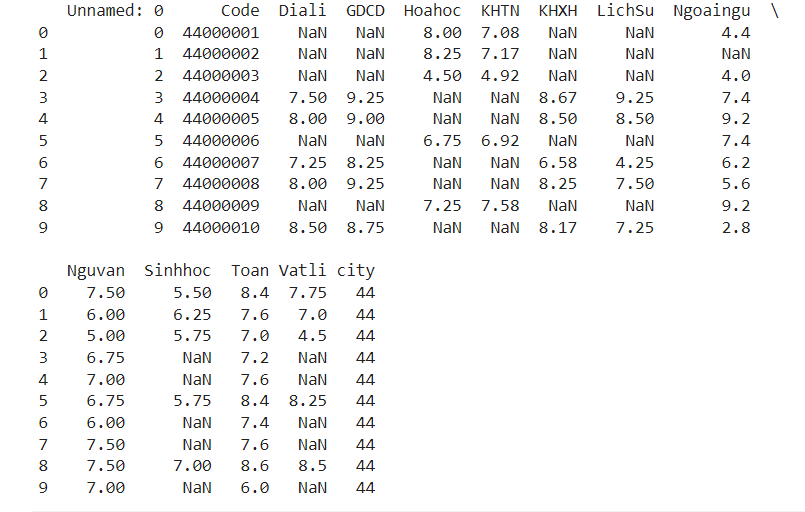


1. **Tiền xử lý dữ liệu:**

* Dữ liệu rất lớn nên không thể xử lý trong Microsoft Excel được, vì thế ta cần dùng thư viện Pandas trong python để hỗ trợ.

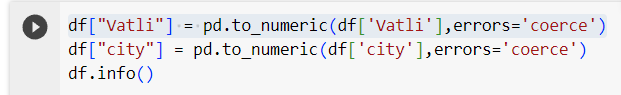


Qua đó, ta xem được có 1.051.860 hàng và 14 cột lưu thông tin điểm thi THPT năm 2020 trên cả nước



*Hình dữ liệu điểm thi của 10 dòng đầu tiên trong tập dữ liệu*

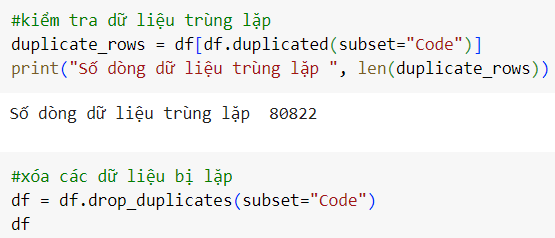
Tiếp theo, ta quan sát kiểu dữ liệu của các cột về điểm số ta cần định dạng các cột là điểm chỉ nhận giá trị là number, mã tỉnh/ thành cũng cần định dạng lại chỉ nhận giá trị number.



errors='coerce': Tham số này xác định cách xử lý khi có giá trị không thể chuyển đổi thành số. Trong trường hợp 'coerce', nếu giá trị không thể chuyển đổi, thì giá trị đó sẽ được đặt là NaN (Not a Number).

Đặt tên lại các cột để đảm bảo dễ theo dõi và đảm bảo đầy đủ tên cột. Tập dữ liệu mới chứa các cột ['No', 'Code', 'Diali', 'GDCD', 'Hoahoc', 'KHTN', 'KHXH', 'LichSu', 'Ngoaingu', 'Nguvan', 'Sinhhoc', 'Toan', 'Vatli', 'City']

Tiến hành kiểm tra và xóa các dòng dữ liệu lặp bằng các dòng bên dưới để tránh thông tin 1 thí sinh lặp lại nhiều hơn 1 lần



Sau khi xóa các dữ liệu đã lặp, ta biết được số thí sinh tham gia dự thi THPT QG, ta có 1051860 hàng tương ứng với 1051860 thí sinh.

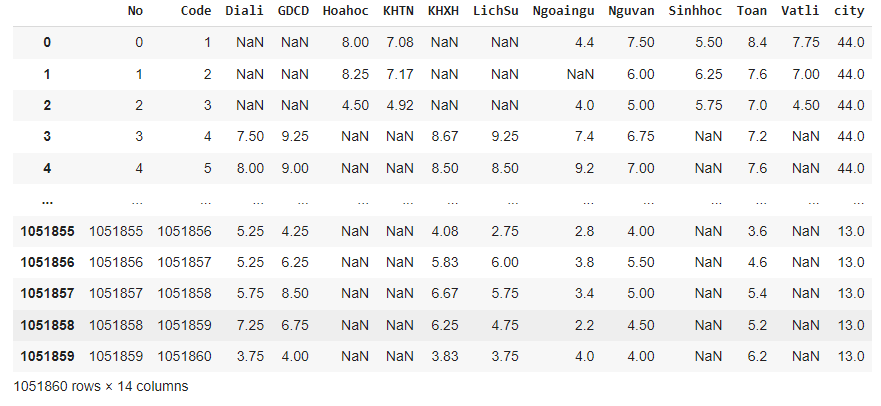
Tiếp theo, đặt tên lại SBD nhằm bảo đảm không lộ lọt thông tin riêng tư.

my\_df = df

my\_df ["Code"] = range (1, len(df)+1)

my\_df

Kết quả:

u

1. **Tách và lọc dữ liệu**

**Đếm số lượng thí sinh không thi cả 2 khối KHTn và KHXH**

**count\_no\_exam = len(my\_df[my\_df["KHTN"].isnull() & my\_df["KHXH"].isnull()])**

**Đếm số lượng thí sinh thi cả 2 khối KHTn và KHXH**

**count\_both\_exam = len(my\_df[my\_df["KHTN"].notnull() & my\_df["KHXH"].notnull()])**

**In kết quả**

**print("Số lượng thí sinh không thi cả 2 môn: ", count\_no\_exam)**

**print("Số lượng thí sinh thi cả 2 môn: ", count\_both\_exam)**

Kết quả:



**Lọc dữ liệu điểm thi của thí sinh tại Tp.HCM**

**df\_khtn\_hcm = my\_df\_KHTN[my\_df\_KHTN["city"]==2]**

**df\_khtn\_hcm.to\_csv ("/content/drive/MyDrive/KHAI PHÁ DỮ LIỆU/THỰC THÀNH 3/df\_khtn\_hcm.csv")**

**Lọc dữ liệu điểm thi của thí sinh tại Hà Nội**

**df\_khtn\_hn = my\_df\_KHTN[my\_df\_KHTN["city"]==1]**

**df\_khtn\_hn.to\_csv ("/content/drive/MyDrive/KHAI PHÁ DỮ LIỆU/THỰC THÀNH 3/df\_khtn\_hn.csv")**

**Tách dataframe my\_df thành 2 dataframe khtn và khxh**

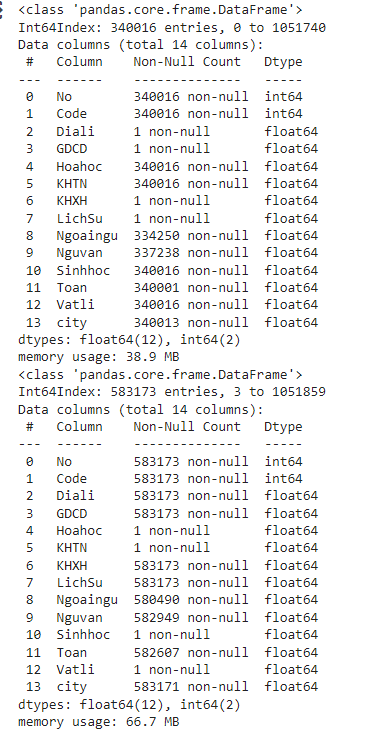
my\_df\_KHTN = my\_df[my\_df["KHTN"].notnull()]

my\_df\_KHXH = my\_df[my\_df["KHXH"].notnull()]

my\_df\_KHTN.info()

my\_df\_KHXH.info()

Kết quả:



1. **Sử dụng K-Mean để thử gom cụm và tìm số cụm có khả năng trong tập dữ liệu**

**Thêm các thư viện cần thiết**

**import matplotlib.pyplot as plt**

**from sklearn.cluster import KMeans**

**from sklearn.metrics import silhouette\_score**

* **KHXH**

**Tiền xử lý dữ liệu**

**my\_df\_KHXH.fillna(my\_df\_KHXH.mean(), inplace=True)**

* **Khoa học tự nhiên**

**df = df.drop\_duplicates(subset="Code")**

**my\_df = df**

**my\_df["Code"] = range(1, len(df) + 1)**

**Tìm số cụm tối ưu sử dụng phương pháp Elbow**

**max\_clusters = 10 # Số cụm tối đa muốn xem**

**sse = []**

**for k in range(1, max\_clusters + 1):**

**kmeans = KMeans(n\_clusters=k, random\_state=42)**

**kmeans.fit(my\_df\_KHXH)**

**sse.append(kmeans.inertia\_)**

* **KHTN**

**Tiền xử lý dữ liệu**

**my\_df\_khtn.fillna(my\_df\_khtn.mean(), inplace=True)**

**Tìm số cụm tối ưu sử dụng phương pháp Elbow**

**max\_clusters = 10 # Số cụm tối đa muốn xem**

**sse = []**

**for k in range(1, max\_clusters + 1):**

**kmeans = KMeans(n\_clusters=k, random\_state=42)**

**kmeans.fit(my\_df\_khtn)**

**sse.append(kmeans.inertia\_)**

1. **Kết quả và nhận xét**

* **KHXH**

**Vẽ biểu đồ phân cụm**

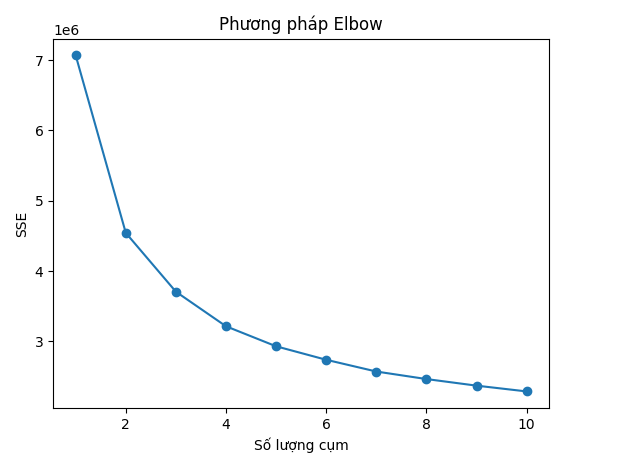
**plt.plot(range(1, max\_clusters + 1), sse, marker='o')**

**plt.xlabel('Số lượng cụm')**

**plt.ylabel('SSE')**

**plt.title('Phương pháp Elbow')**

**plt.show()**

****

Quan sát biểu đồ, ta thấy số lượng tối ưu có thể là 5

Tiếp theo, áp dụng thuật toán K-means để vẽ biểu đồ phân cụm

**Số cụm tối ưu dựa trên biểu đồ Elbow**

num\_clusters = 5

**Áp dụng thuật toán K-means**

kmeans = KMeans(n\_clusters=num\_clusters, random\_state=42)

kmeans.fit(my\_df\_KHXH)

**Gán nhãn cụm cho dữ liệu**

my\_df\_KHXH['Cluster'] = kmeans.labels\_

**Vẽ biểu đồ phân cụm**

plt.scatter(my\_df\_KHXH['Cluster'], my\_df\_KHXH['Toan'], c=my\_df\_KHXH['Cluster'], cmap='viridis')

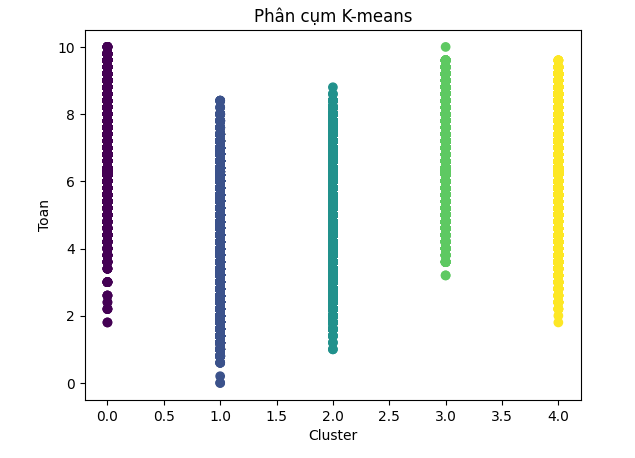
plt.xlabel('Cluster')

plt.ylabel('Toan')

plt.title('Phân cụm K-means')

plt.show()

**Kết quả:**

****

Quan sát biểu đồ ta thấy, cụm 1 và cụm 5 có sự phân hóa rõ rệt từ điểm 2 đến 10, trong khi đó cụm 3 chỉ phân bố từ 4 điểm đến 10 điểm

Bên cạnh đó, cụm 3 và cụm 3 lại có sự phân hóa rõ rệt từ 2 điểm đến 8 điểm.

* **KHTN**

**Vẽ biểu đồ phân cụm**

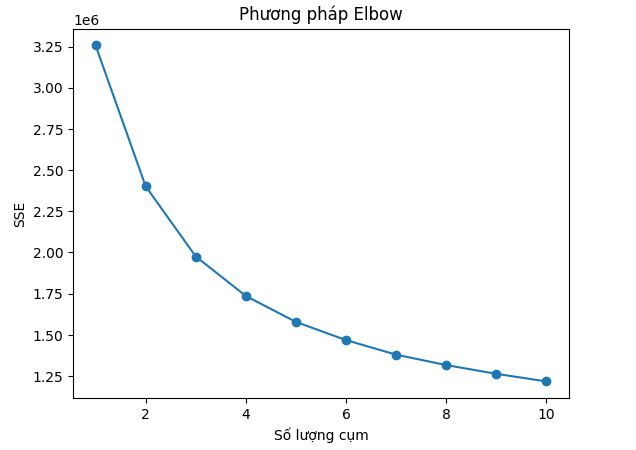
**plt.scatter(my\_df\_khtn['Cluster'], my\_df\_khtn['Toan'], c=my\_df\_khtn['Cluster'], cmap='viridis')**

**plt.xlabel('Cluster')**

**plt.ylabel('Toan')**

**plt.title('Phân cụm K-means')**

**plt.show()**

****

Quan sát biểu đồ, ta thấy số lượng tối ưu có thể là 5

Tiếp theo, áp dụng thuật toán K-means để vẽ biểu đồ phân cụm

**Số cụm tối ưu dựa trên biểu đồ Elbow**

num\_clusters = 5

**Áp dụng thuật toán K-means**

kmeans = KMeans(n\_clusters=num\_clusters, random\_state=42)

kmeans.fit(my\_df\_KHTN)

**Gán nhãn cụm cho dữ liệu**

my\_df\_KHTN['Cluster'] = kmeans.labels\_

**Vẽ biểu đồ phân cụm**

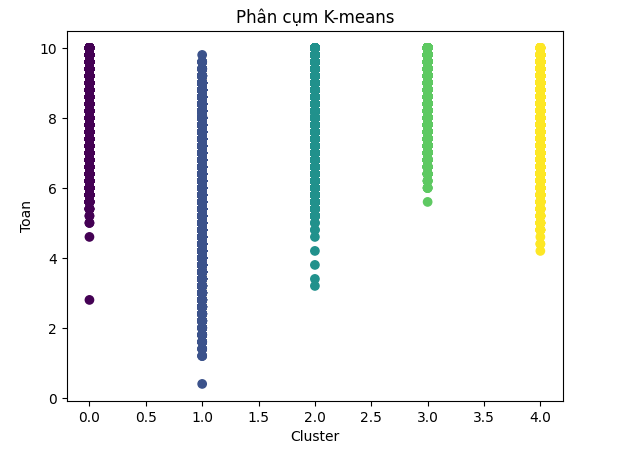
plt.scatter(my\_df\_KHTN['Cluster'], my\_df\_KHTN['Toan'], c=my\_df\_KHTN['Cluster'], cmap='viridis')

plt.xlabel('Cluster')

plt.ylabel('Toan')

plt.title('Phân cụm K-means')

plt.show()



Quan sát biểu đồ ta thấy, cụm 0 và cụm 3 có sự phân hóa rõ rệt từ điểm 4 đến 10, trong khi đó cụm 1 chỉ phân bố rải rác từ 0 điểm đến 10 điểm

Bên cạnh đó, cụm 2 và cụm 4 lại có sự phân hóa rõ rệt từ 4 điểm đến 10 điểm

Vẽ biểu đồ Elbow sử dụng thuật toán Kmeans

**khtn**

plt.plot(X0[:, 0], X0[:, 1], 'b^', markersize = 4, alpha = .8)

plt.plot(X1[:, 0], X1[:, 1], 'go', markersize = 4, alpha = .8)

plt.plot(X2[:, 0], X2[:, 1], 'rs', markersize = 4, alpha = .8)

# Số cụm tối ưu dựa trên biểu đồ Elbow

num\_clusters = 5

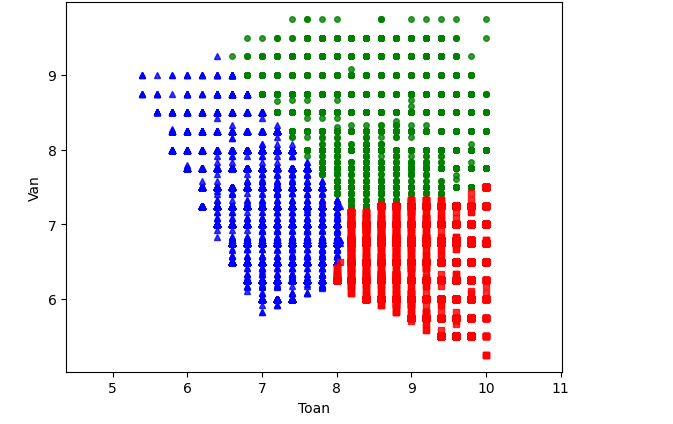
# Áp dụng thuật toán K-means

kmeans = KMeans(n\_clusters=num\_clusters, random\_state=0).fit(my\_df\_khtn[["Toan", "Nguvan"]])

print("Center found by scikit-learn:\n", kmeans.cluster\_centers\_)

pred\_label = kmeans.predict(my\_df\_khtn[["Toan", "Nguvan"]])

kmeans\_display(my\_df\_khtn[["Toan", "Nguvan"]].values, pred\_label)



Xét biểu đồ trên, ta thấy học giỏi môn văn được phân hóa mạnh hơn so với toán, số cụm phân hóa giỏi văn cũng sẽ giỏi toán được phân hóa rời rạc. Có thể kết luận, số lượng học sinh học giỏi toán và văn là khá ít.

**khxh**