ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN
KHOA KHOA HỌC VÀ KỸ THUẬT THÔNG TIN

BÁO CÁO BÀI TẬP
KNN – K NEAREST NEIGHBORS

Giảng viên hướng dẫn: ThS. Huỳnh Văn Tín, TS. Nguyễn Văn Kiệt.

Sinh viên thực hiện:

23520993 **Nguyễn Thảo Nga**

23521033 **Trần Khánh Ngọc**

23521131 **Trần Thị Hoàng Nhung**

23521204 **Nguyễn Đặng Quang Phúc**

Thành phố Hồ Chí Minh, tháng 10 năm 2025

MỤC LỤC

1. Giới thiệu

Thuật toán **k-Nearest Neighbors (kNN)** là một trong những phương pháp học có giám sát (*supervised learning*) đơn giản và hiệu quả, được ứng dụng rộng rãi trong khai phá dữ liệu và học máy.
Ý tưởng cốt lõi của kNN là **các điểm dữ liệu tương tự nhau có xu hướng nằm gần nhau trong không gian đặc trưng**.
Thuật toán này được gọi là **lazy learning**, vì mô hình không học trước mà chỉ thực hiện tính toán khi cần dự đoán.

2. Lý thuyết về thuật toán kNN

2.1. Khái niệm

kNN là thuật toán **phân loại dựa trên khoảng cách** giữa các điểm dữ liệu.
Khi cần phân loại một điểm dữ liệu mới, mô hình sẽ tìm **k điểm lân cận gần nhất** trong tập dữ liệu huấn luyện, sau đó dự đoán nhãn của điểm mới dựa trên **đa số nhãn của các điểm lân cận này**.

2.2. Ý tưởng hoạt động

Thuật toán giả định rằng **các dữ liệu có đặc trưng tương tự sẽ nằm gần nhau trong không gian đặc trưng**.
Việc đo khoảng cách giữa hai điểm có thể được thực hiện bằng nhiều công thức khác nhau, phổ biến nhất là:

- Khoảng cách **Euclidean**
- Khoảng cách **Manhattan**
- Khoảng cách **Minkowski**

2.3. Công thức tính khoảng cách

Khoảng cách giữa hai điểm dữ liệu x và y có k thuộc tính được tính bằng công thức:

Euclidean $\sqrt{\sum_{i=1}^{k}(x_i - y_i)^2}$

Manhattan $\sum_{i=1}^{k}|x_i - y_i|$

Minkowski $\left(\sum_{i=1}^{k}(|x_i - y_i|)^q\right)^{1/q}$

trong đó **xi** và **yi** lần lượt là các giá trị thuộc tính của hai điểm dữ liệu.

2.4. Các bước thực hiện thuật toán kNN

1. Chuẩn bị tập dữ liệu huấn luyện D (đã có nhãn) và tập dữ liệu kiểm định A (chưa có nhãn).
2. Tính khoảng cách từ mỗi điểm dữ liệu trong A đến tất cả các điểm trong D.
3. Chọn **k điểm gần nhất** (k là siêu tham số cần chọn phù hợp).
4. Đếm số lượng nhãn xuất hiện trong k điểm này.
5. Gán nhãn cho điểm kiểm định dựa trên lớp xuất hiện nhiều nhất.

2.5. Ưu và nhược điểm

Ưu điểm:

- Đơn giản, dễ hiểu, dễ triển khai.
- Không cần giả định về phân phối dữ liệu.
- Hoạt động tốt với bài toán **phân loại đa lớp**.
- Có thể sử dụng cho cả **phân loại (classification)** và **hồi quy (regression)**.

Nhược điểm:

- Chậm khi kích thước dữ liệu lớn.
- Tốn bộ nhớ vì phải lưu toàn bộ tập huấn luyện.
- Nhạy cảm với nhiễu và dữ liệu không cân bằng.

3. Ứng dụng của thuật toán kNN

- **Y tế:** Chẩn đoán bệnh dựa trên dữ liệu bệnh nhân cũ hoặc đề xuất phác đồ điều trị tương tự.
- **Ngân hàng:** Dự đoán rủi ro tín dụng, phát hiện gian lận giao dịch dựa trên lịch sử giao dịch.
- **Giáo dục:** Phân loại học sinh theo năng lực hoặc hoàn cảnh để đưa ra hình thức hỗ trợ phù hợp.
- **Thương mại điện tử:** Xây dựng **hệ thống gợi ý sản phẩm (recommendation system)**, cá nhân hóa trải nghiệm người dùng.
- **Kinh tế:** Dự báo xu hướng thị trường, dự đoán biến động giá cổ phiếu hoặc tình hình kinh tế vĩ mô.

4. Phân tích Dữ liệu thăm dò (EDA) Bộ dữ liệu UIT-VSFC

4.1. Giới thiệu

Bộ dữ liệu **UIT-VSFC (Vietnamese Students' Feedback Corpus)** là một nguồn tài nguyên miễn phí và chất lượng cao, được xây dựng nhằm phục vụ nghiên cứu **phân tích cảm xúc tiếng Việt**, đặc biệt trong lĩnh vực **giáo dục**.

4.2. Thu thập và Tiền xử lý Dữ liệu

1. Nguồn gốc và Quy mô

Phản hồi của sinh viên được thu thập từ một trường đại học tại Việt Nam thông qua các cuộc khảo sát cuối mỗi học kỳ, giai đoạn **2013 – 2016**.
Bộ dữ liệu cuối cùng bao gồm **hơn 16.000 câu** đã được gán nhãn về **cảm xúc** và **chủ đề**.

Thống kê dữ liệu thu thập theo năm học:

Năm học	Số Giảng viên	Số Sinh viên	Số Môn học	Số Phản hồi
2014 – 2015	175	2 235	143	6 038
2015 – 2016	184	2 856	160	6 288

2016 – 2017	227	3 789	175	13 417

2. Đặc điểm phản hồi và tiền xử lý

Phản hồi của sinh viên Việt Nam thường ngắn gọn, mang tính tự do và có thể chứa nhiều **viết tắt**, **lỗi chính tả**, **biểu tượng cảm xúc**, hoặc **ký tự đặc biệt**.

- **Viết tắt:** gv (giảng viên), sv (sinh viên), hk1 (học kỳ 1), …
- **Biểu tượng cảm xúc:** tích cực (<3, :-), :D) ; tiêu cực (:()
- **Quy trình tiền xử lý:**
    1. Phân đoạn phản hồi thành câu.
    2. Thay thế viết tắt và sửa lỗi chính tả bằng từ điển chuẩn.
    3. Ẩn danh thông tin cá nhân để bảo mật.
    4. Thay thế biểu tượng cảm xúc bằng chuỗi chữ (vd. :D → colonsmile, <3 → colonlove).

4.3. Cấu trúc Gán nhãn Dữ liệu

1. Gán nhãn Cảm xúc (Sentiment-based Task)

Mỗi câu được gán **một trong ba cực cảm xúc** sau:

Mã	Nhãn Cảm xúc	Mô tả
2	**Tích cực (Positive)**	Thể hiện sự hài lòng hoặc khen ngợi.
0	**Tiêu cực (Negative)**	Thể hiện sự phàn nàn hoặc không hài lòng.
1	**Trung lập (Neutral)**	Không thể hiện rõ ý kiến hoặc không hoàn chỉnh về nghĩa.

Lưu ý: Với các câu chứa cả hai cực cảm xúc (thường có từ nối như “nhưng”, “tuy nhiên”), người gán nhãn chọn cực cảm xúc mạnh hơn.

2. Gán nhãn Chủ đề (Topic-based Task)

Mã	Nhãn Chủ đề	Mô tả
0	**Giảng viên (Lecturer)**	Liên quan đến phương pháp, thái độ, kiến thức của giảng viên.
1	**Chương trình học (Curriculum)**	Liên quan đến nội dung, bài tập, điểm số, cấu trúc môn học.
2	**Cơ sở vật chất (Facility)**	Liên quan đến phòng học, thiết bị, ánh sáng, máy chiếu, quạt, …
3	**Khác (Others)**	Không thuộc các nhóm chủ đề trên.

4.4. Phân tích Phân phối Dữ liệu

1. Phân phối Cảm xúc và Chủ đề

Phân phối	Tích cực	Tiêu cực	Trung lập	Tổng cộng
Giảng viên (71.76%)	33.57%	25.38%	1.81%	71.76%
Chương trình học (18.79%)	3.40%	14.39%	1.00%	18.79%
Cơ sở vật chất (4.40%)	0.11%	4.21%	0.08%	4.40%
Khác (5.04%)	1.61%	2.01%	1.43%	5.04%
Tổng cộng (%)	**49.69**	**45.99**	**4.32**	**100**

Nhận xét:

- **Cảm xúc:** dữ liệu **mất cân bằng**, chủ yếu là **Tích cực (49.69%)** và **Tiêu cực (45.99%)**, trong khi **Trung lập chỉ 4.32%**.
- **Chủ đề:** phản hồi tập trung nhiều vào **Giảng viên (71.76%)**, gây mất cân bằng cao.

2. Phân tích Độ dài Câu

- Phần lớn phản hồi ngắn (**1 – 15 từ**, chiếm hơn 83%).
- Câu **Tiêu cực** thường **dài hơn** vì chứa lý do hoặc đề xuất giải pháp.
- Phản hồi về **Giảng viên**, **Chương trình học**, và **Cơ sở vật chất** có xu hướng **dài hơn năm từ** để mô tả chi tiết hơn.

5. Cài đặt và Đánh giá Mô hình KNN

5.1. Môi trường và Thư viện

Thí nghiệm được thực hiện bằng ngôn ngữ **Python 3.10**, sử dụng các thư viện:

- **pandas**: xử lý dữ liệu dạng bảng (CSV, Excel).
- **re (Regular Expressions)**: xử lý và thay thế các ký tự đặc biệt trong văn bản.
- **scikit-learn (sklearn)**: xây dựng mô hình học máy, bao gồm:
  - TfidfVectorizer: chuyển văn bản thành vector TF-IDF.
  - KNeighborsClassifier: cài đặt thuật toán kNN.
  - accuracy_score, f1_score, classification_report: đánh giá mô hình.
- **warnings** và **openpyxl**: hỗ trợ xử lý cảnh báo và đọc tệp .xlsx.

5.2. Tiền xử lý Dữ liệu

Bước 1. Tải dữ liệu

Bộ dữ liệu được chia thành ba phần:

- **Train.xlsx** – tập huấn luyện (11.426 mẫu)
- **Dev.xlsx** – tập phát triển (validation)
- **Test.xlsx** – tập kiểm thử (3.166 mẫu)

Dữ liệu được đọc bằng pandas.read_excel() với cấu trúc:

- Sents: cột chứa câu phản hồi (dữ liệu văn bản).

- Sentiments: nhãn cảm xúc (0 – Tiêu cực, 1 – Trung lập, 2 – Tích cực).

Bước 2. Xử lý giá trị NaN và tiền xử lý văn bản

Các bước chính:

1. Loại bỏ hàng bị thiếu (NaN) trong cột *Sents* hoặc *Sentiments*.
2. Xây dựng **hàm preprocess_text()** để thay thế các ký hiệu và biểu tượng cảm xúc dựa trên file *README.txt*.
   Ví dụ:
   - :) → colonsmile, :(→ colonsad, <3 → colonlove
   - @@ → colonsurprise, :v → colonbigsmile, c# → csharp
3. Hàm sử dụng re.sub() để thay thế theo biểu thức chính quy (regex), đồng thời chuẩn hóa và loại bỏ dấu câu, ký tự đặc biệt.
4. Tạo cột mới Sents_processed để lưu văn bản đã tiền xử lý.

Kết quả: Dữ liệu đầu vào được làm sạch và chuẩn hóa, sẵn sàng cho quá trình vector hóa.

5.3. Biểu diễn Dữ liệu bằng TF-IDF

Văn bản được chuyển thành dạng số bằng **TF-IDF (Term Frequency – Inverse Document Frequency)**.

Công cụ:

tfidf_vectorizer = TfidfVectorizer(max_features=5000)

- max_features=5000: chỉ giữ lại 5000 từ phổ biến nhất trong tập huấn luyện, giúp giảm chiều dữ liệu và loại bỏ nhiễu.
- **Quy trình:**
  - Gọi .fit_transform() trên tập **Train** để học từ vựng và biến đổi văn bản.
  - Gọi .transform() trên **Dev** và **Test** để đảm bảo cùng không gian đặc trưng.

Kích thước ma trận TF-IDF thu được:

Train: (11,426 × 2,459)

5.4. Tối ưu Siêu tham số K (Hyperparameter Tuning)

Giá trị **K** (số lượng hàng xóm gần nhất) được chọn bằng cách thử nghiệm trên tập **Dev** với các giá trị lẻ để tránh hòa phiếu:

K	1	3	5	7	9	11	15	21
F1-score (Dev)	0.7914	0.8017	0.8204	**0.8309**	0.8254	0.8235	0.8239	0.8199

Kết quả:

K tối ưu = 7, đạt **F1-score = 0.8309** trên tập Dev.

Thuật toán được cài đặt bằng:

knn = KNeighborsClassifier(n_neighbors=k)

knn.fit(X_train_tfidf, y_train)

Trong đó:

- Mỗi mẫu huấn luyện được lưu trữ trong không gian vector TF-IDF.
- Khi dự đoán, thuật toán tính **khoảng cách Euclidean** giữa mẫu cần phân loại và tất cả các mẫu huấn luyện, rồi lấy đa số phiếu của k hàng xóm gần nhất.

5.5. Huấn luyện và Đánh giá trên Tập Test

Huấn luyện mô hình cuối cùng:

final_knn = KNeighborsClassifier(n_neighbors=7)

final_knn.fit(X_train_tfidf, y_train)

Mô hình được huấn luyện lại trên toàn bộ tập Train (với K=7) và đánh giá trên tập Test.

Kết quả tổng quan:

Chỉ số	Giá trị
Accuracy	**0.8206 (82.06%)**
F1-score (Weighted)	**0.8087**

5.6. Báo cáo Chi tiết (Classification Report)

Lớp	Precision	Recall	F1-score	Support
Class 0 (Negative)	0.83	0.82	0.82	1,409
Class 1 (Neutral)	0.48	0.14	0.22	167
Class 2 (Positive)	0.82	0.90	0.86	1,590
Accuracy (Tổng)			**0.82**	3,166
Macro Avg	0.71	0.62	0.63	3,166
Weighted Avg	0.81	0.82	0.81	3,166

5.7. Phân tích và Nhận xét

- **Hiệu quả mô hình:**
  Mô hình **KNN (K=7)** đạt độ chính xác cao (**82.06%**) và F1-score tổng thể tốt (**0.81**).
  Điều này cho thấy khả năng mô hình học được đặc trưng cảm xúc từ dữ liệu văn bản tiếng Việt sau tiền xử lý.

- **Vấn đề mất cân bằng dữ liệu:**
  Lớp **Trung lập (Class 1)** có F1 thấp (**0.22**) do số lượng mẫu ít, dẫn đến độ phủ thấp (Recall chỉ 0.14).
- **Nguyên nhân và hướng cải thiện:**
  1. Dữ liệu mất cân bằng mạnh → có thể áp dụng **SMOTE** hoặc **class weighting**.
  2. TF-IDF tạo không gian vector cao chiều → có thể dùng **PCA** hoặc **TruncatedSVD** để giảm chiều.
  3. KNN không học tham số, nên nhạy cảm với nhiễu → có thể so sánh với mô hình tuyến tính như **Logistic Regression** hoặc **SVM** để tăng độ ổn định.

6. Slide: Canva

7. Code: Google Collab