ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC BÁCH KHOA

BÁO CÁO BÀI TẬP LỚN

MÁY HỌC CƠ BẢN VÀ ỨNG DỤNG (EE3169)

PHÂN TÍCH HÀNH VI NGƯỜI DÙNG THIẾT BỊ DI ĐỘNG BẰNG MÔ HÌNH KNN

Lớp - Nhóm: L01 - Nhóm 17

Giảng viên hướng dẫn: ThS. Nguyễn Khánh Lợi

Sinh viên thực hiện: Phạm Khánh MSSV: 2011391

Võ Văn Trí Anh MSSV: 2010137

Phan Vũ Bảo Tín MSSV: 2010701

Tp. Hồ Chí Minh – 2024

Mục lục

1 Mở đầu

1.1 Lý do chọn đề tài

Trong thời đại công nghệ số hiện nay, việc sử dụng thiết bị di động đã trở thành một phần không thể thiếu trong cuộc sống hàng ngày của con người. Từ việc liên lạc, làm việc, giải trí cho đến việc quản lý sức khỏe, các ứng dụng di động đã góp phần cải thiện hiệu suất và tiện ích trong nhiều lĩnh vực. Đồng thời, số lượng người sử dụng thiết bị di động ngày càng tăng lên, tạo ra một lượng lớn dữ liệu người dùng với các thông tin phong phú, đa dạng như thời gian sử dụng, thói quen tương tác, vị trí địa lý, và các hành vi trong suốt quá trình sử dụng.

Trong bối cảnh đó, phân tích hành vi người dùng thiết bị di động trở thành một lĩnh vực nghiên cứu thú vị và đầy tiềm năng. Việc hiểu rõ hơn về hành vi người dùng không chỉ giúp tối ưu hóa các ứng dụng, nâng cao trải nghiệm người dùng mà còn mở ra cơ hội cho việc phát triển các ứng dụng cá nhân hóa, cải thiện dịch vụ khách hàng, và đề xuất những sản phẩm phù hợp hơn.

Máy học, với khả năng xử lý và phân tích một lượng lớn dữ liệu, đã trở thành công cụ mạnh mẽ trong việc khai thác thông tin từ các hành vi người dùng. Các thuật toán máy học có thể tự động nhận diện các mẫu hành vi, phát hiện xu hướng và đưa ra những dự đoán chính xác về nhu cầu và thói quen của người dùng. Do đó, nghiên cứu ứng dụng máy học để phân tích hành vi người dùng thiết bị di động không chỉ có ý nghĩa về mặt lý thuyết mà còn có giá trị thực tiễn cao trong việc tối ưu hóa trải nghiệm người dùng và phát triển các giải pháp công nghệ mới.

Với những lý do trên, nhóm chọn nghiên cứu đề tài **"Phân tích hành vi người dùng thiết bị di động bằng máy học dựa trên mức độ sử dụng"** nhằm khám phá khả năng ứng dụng máy học trong việc phân tích và hiểu rõ hơn về hành vi người dùng thiết bị di động. Đây là một lĩnh vực đang phát triển nhanh chóng và có nhiều ứng dụng tiềm năng trong thực tiễn, đồng thời góp phần nâng cao chất lượng các sản phẩm và dịch vụ công nghệ.

1.2 Mục tiêu nghiên cứu

Mục tiêu chính của đề tài này là áp dụng các phương pháp máy học để phân tích hành vi người dùng thiết bị di động, đặc biệt dựa trên mức độ sử dụng thiết bị trong các tình huống và thời gian khác nhau. Dữ liệu nghiên cứu được lấy từ bộ dữ liệu *Mobile Device Usage and User Behavior trên Kaggle* (https://www.kaggle.com/datasets/valakhorasani/mobile-device-usage-and-user-behavior-dataset), bao gồm thông tin về thói quen sử dụng thiết bị di động

của người dùng. Cụ thể, các mục tiêu nghiên cứu của bài tập lớn này bao gồm:

1. **Xác định các đặc trưng hành vi người dùng từ dữ liệu thiết bị di động**: Dựa trên bộ dữ liệu thu thập từ Kaggle, nhóm đề tài sẽ phân tích các đặc trưng hành vi người dùng như
2. **Ứng dụng các thuật toán máy học để phân tích hành vi người dùng**: Nghiên cứu sẽ áp dụng các kỹ thuật máy học như phân loại, phân cụm, và học sâu để nhận diện các mẫu hành vi của người dùng dựa trên mức độ sử dụng thiết bị. Việc này nhằm mục tiêu phát hiện các xu hướng và nhóm người dùng có thói quen tương tự, đồng thời dự đoán hành vi người dùng trong tương lai.
3. **Đánh giá hiệu quả của các mô hình máy học trong việc phân tích hành vi**: So sánh và đánh giá hiệu quả của các thuật toán máy học khác nhau, từ đó chọn ra các mô hình chính xác và hiệu quả nhất trong việc phân tích hành vi người dùng thiết bị di động, dựa trên bộ dữ liệu đã thu thập.
4. **Đề xuất các ứng dụng thực tiễn cho ngành công nghệ di động**: Dựa trên các kết quả phân tích, nhóm đề tài sẽ đưa ra những ứng dụng thực tế trong việc tối ưu hóa trải nghiệm người dùng, phát triển các ứng dụng cá nhân hóa và cải thiện các dịch vụ di động dựa trên hành vi người dùng.
5. **Khám phá các vấn đề và thách thức trong việc áp dụng máy học vào phân tích hành vi người dùng thiết bị di động**: Phân tích những thách thức trong việc xử lý và làm sạch dữ liệu, cũng như những vấn đề phát sinh khi áp dụng các mô hình máy học trong môi trường thực tế. Đồng thời, đề xuất các hướng cải tiến trong việc triển khai các mô hình máy học để đạt được kết quả tối ưu.

Thông qua các mục tiêu nghiên cứu trên, nghiên cứu này hy vọng sẽ cung cấp những thông tin giá trị trong việc phân tích hành vi người dùng thiết bị di động, từ đó đóng góp vào việc phát triển các ứng dụng và dịch vụ công nghệ phục vụ nhu cầu người sử dụng.

1.3 Phạm vi và đối tượng nghiên cứu

1.3.1 Phạm vi nghiên cứu

Nghiên cứu này tập trung vào việc phân tích hành vi người dùng thiết bị di động thông qua việc áp dụng các phương pháp máy học đối với dữ liệu thu thập từ bộ dữ liệu *Mobile Device Usage and User Behavior* có sẵn trên Kaggle. Bộ dữ liệu này bao gồm thông tin về mức độ sử dụng các ứng dụng di động, thời gian sử dụng thiết bị, các loại ứng dụng và thói quen tương tác của người dùng trong một khoảng thời gian nhất định. Phạm vi nghiên cứu sẽ giới hạn trong việc phân tích các mẫu hành vi từ các đặc trưng dữ liệu có

sẵn, sử dụng các thuật toán máy học để phân tích hành vi người dùng và đưa ra những dự đoán về xu hướng sử dụng thiết bị di động trong tương lai.

1.3.2 Đối tượng nghiên cứu

Đối tượng nghiên cứu của tiểu luận này là hành vi của người dùng thiết bị di động, cụ thể là các mẫu hành vi liên quan đến mức độ sử dụng thiết bị, thói quen sử dụng ứng dụng, và các yếu tố ảnh hưởng đến hành vi người dùng (như thời gian, tần suất và mục đích sử dụng). Dữ liệu được thu thập từ bộ dữ liệu có sẵn trên Kaggle, bao gồm các thông tin về các nhóm người dùng di động trong các tình huống và môi trường khác nhau. Đối tượng nghiên cứu không bao gồm các yếu tố khác như đặc điểm nhân khẩu học hay các yếu tố ngoại vi khác không có trong bộ dữ liệu.

1.3.3 Giới hạn của nghiên cứu

Dữ liệu nghiên cứu được lấy từ bộ dữ liệu đã có sẵn, và do đó, tính chính xác của các kết quả phụ thuộc vào chất lượng và tính đại diện của bộ dữ liệu này.

Đề tài không xem xét các yếu tố ngoại cảnh ngoài dữ liệu thu thập được (chẳng hạn như tác động của các yếu tố xã hội, văn hóa đến hành vi người dùng), mà chỉ tập trung vào các hành vi người dùng có thể đo lường được thông qua các chỉ số sử dụng thiết bị.

2 Phương pháp nghiên cứu

2.1 Dữ liệu sử dụng

2.1.1 Mô tả nguồn dữ liệu

Bộ dữ liệu được lấy từ **Kaggle** (https://www.kaggle.com/datasets/valakhorasani/mobile-device-usage-and-user-behavior-dataset/) với tiêu đề **Mobile Device Usage and User Behavior Dataset**. Đây là một tập hợp dữ liệu gồm 700 bản ghi, mỗi bản ghi đại diện cho một người dùng thiết bị di động. Dữ liệu này được thiết kế để nghiên cứu hành vi sử dụng thiết bị di động của người dùng dựa trên nhiều đặc trưng như thời gian sử dụng ứng dụng, thời gian bật màn hình, mức độ tiêu thụ pin, và thông tin nhân khẩu học.

2.1.2 Các đặc trưng của dữ liệu

Dữ liệu bao gồm 11 cột, được mô tả chi tiết như sau:

Tên đặc trưng	Kiểu dữ liệu	Mô tả
User ID	Integer	Mã định danh duy nhất cho mỗi người dùng.
Device Model	String	Mẫu thiết bị di động (ví dụ: Google Pixel 5, iPhone 12).
Operating System	String	Hệ điều hành của thiết bị (Android hoặc iOS).
App Usage Time (min/day)	Integer	Tổng thời gian sử dụng ứng dụng trong ngày (đơn vị: phút).
Screen On Time (hours/day)	Float	Thời gian bật màn hình thiết bị trong ngày (đơn vị: giờ).
Battery Drain (mAh/day)	Integer	Lượng pin tiêu hao trung bình mỗi ngày (đơn vị: mAh).
Number of Apps Installed	Integer	Số lượng ứng dụng đã cài đặt trên thiết bị.
Data Usage (MB/day)	Integer	Lượng dữ liệu đã sử dụng trung bình mỗi ngày (đơn vị: MB).

Age	Integer	Độ tuổi của người dùng.
Gender	String	Giới tính của người dùng (Male/Female).
User Behavior Class	Integer	Nhãn phân loại hành vi người dùng (1, 2, 3, 4) dựa trên mức độ sử dụng.

2.2 Lựa chọn mô hình máy học

Trong bài tập lớn này, việc sử dụng mô hình **K-Nearest Neighbors (KNN)** là hoàn toàn phù hợp vì nhiều lý do. Trước hết, KNN là một mô hình học máy đơn giản và dễ hiểu, dựa trên ý tưởng cơ bản "tìm neighbor gần nhất". Mô hình này hoạt động bằng cách phân loại hoặc dự đoán điểm dữ liệu mới dựa trên khoảng cách với các điểm lân cận trong không gian đặc trưng. Điều này giúp người đọc dễ dàng hiểu và trực quan hóa cách mô hình đưa ra quyết định, phù hợp với mục tiêu của bài nghiên cứu.

Một lợi thế lớn của KNN là nó không yêu cầu các giả định về phân phối dữ liệu, do đó được xếp vào nhóm các mô hình phi tham số. Điều này rất quan trọng trong trường hợp dữ liệu về hành vi người dùng thiết bị di động, bởi vì các đặc trưng như thời gian sử dụng thiết bị hay mức tiêu hao pin không nhất thiết phải tuân theo các phân phối tuyến tính hoặc chuẩn. Nhờ vậy, KNN có khả năng xử lý tốt dữ liệu có mối quan hệ phi tuyến tính giữa các đặc trưng.

Hơn nữa, KNN hoạt động rất hiệu quả trong các bài toán phân loại. Trong nghiên cứu này, mục tiêu chính là phân loại hành vi người dùng dựa trên các đặc trưng như thời gian sử dụng thiết bị hay tuổi của người dùng. Với nhãn phân loại rõ ràng, KNN có thể tìm ra mối quan hệ giữa các đặc trưng và đưa ra kết quả phân loại chính xác. Đặc biệt, với tập dữ liệu kích thước vừa phải như trong bài nghiên cứu, KNN không gặp vấn đề về hiệu suất và có thể tận dụng toàn bộ dữ liệu để huấn luyện và kiểm tra.

Một yếu tố khác khiến KNN trở thành lựa chọn hợp lý là khả năng điều chỉnh linh hoạt thông qua siêu tham số `k` (số neighbor gần nhất). Giá trị của `k` đóng vai trò quan trọng trong hiệu quả của mô hình. Khi `k` nhỏ, mô hình nhạy hơn với các điểm dữ liệu gần, nhưng dễ bị ảnh hưởng bởi nhiễu. Ngược lại, khi `k` lớn, mô hình tổng quát hóa tốt hơn nhưng có thể bỏ qua các chi tiết nhỏ trong dữ liệu. Nhờ sử dụng các công cụ tối ưu hóa như `RandomizedSearchCV`, bài nghiên cứu có thể tìm ra giá trị `k` tối ưu để đạt hiệu suất tốt nhất.

Ngoài ra, KNN là mô hình dễ mở rộng và giải thích. Nếu cần thêm các đặc trưng khác như số lượng ứng dụng cài đặt hay loại ứng dụng được sử dụng nhiều nhất, KNN vẫn hoạt động tốt mà không cần thay đổi cấu trúc. Kết quả của mô hình cũng dễ dàng trực quan hóa, giúp minh họa cách các điểm dữ liệu mới được gán nhãn dựa trên khoảng cách

với các điểm lân cận. Điều này không chỉ giúp người viết hiểu rõ mô hình mà còn hỗ trợ người đọc trong việc hình dung kết quả.

Tuy nhiên, KNN cũng có một số hạn chế. Mô hình nhạy cảm với dữ liệu không cân bằng, ví dụ khi một lớp chiếm ưu thế hơn hẳn các lớp khác. Để khắc phục, có thể chuẩn hóa dữ liệu hoặc sử dụng các phương pháp làm cân bằng lớp. Ngoài ra, với các tập dữ liệu lớn, việc tính toán khoảng cách giữa các điểm có thể tốn tài nguyên. Dẫu vậy, với tập dữ liệu vừa phải như trong bài nghiên cứu này, vấn đề này không đáng lo ngại.

Nhìn chung, mô hình KNN là một lựa chọn phù hợp cho bài toán này. Sự đơn giản, linh hoạt, và khả năng hoạt động tốt trên dữ liệu phân loại như hành vi người dùng khiến KNN trở thành một công cụ hiệu quả trong việc đạt được mục tiêu nghiên cứu. Nếu cần mở rộng thêm hoặc so sánh với các mô hình khác, KNN vẫn là một tiêu chuẩn cơ bản để tham chiếu.

3 Cơ sở lý thuyết

3.1 Giới thiệu về máy học

Máy học (Machine Learning) là một nhánh của trí tuệ nhân tạo (AI), tập trung vào việc xây dựng các hệ thống và mô hình có khả năng tự động học hỏi từ dữ liệu và cải thiện hiệu suất mà không cần lập trình rõ ràng cho từng tình huống cụ thể. Máy học cho phép các hệ thống không chỉ xử lý dữ liệu mà còn phát hiện các mẫu, xu hướng và mối quan hệ ẩn trong dữ liệu, từ đó đưa ra dự đoán hoặc quyết định chính xác hơn.

Máy học hoạt động dựa trên việc sử dụng các thuật toán để huấn luyện mô hình từ dữ liệu. Khi cung cấp dữ liệu đầu vào, mô hình sẽ học cách nhận diện các mẫu và đưa ra kết quả tương ứng. Càng nhiều dữ liệu chất lượng cao, mô hình càng học tốt hơn và cải thiện độ chính xác trong các bài toán thực tế.

Các loại máy học chính

Máy học được chia thành ba loại chính, dựa trên cách thức mô hình học từ dữ liệu:

1. Học có giám sát (Supervised Learning):

- Mô hình được huấn luyện trên dữ liệu đã gắn nhãn, trong đó đầu vào (input) và đầu ra (output) đã được xác định rõ.
- Ví dụ: Dự đoán thời gian sử dụng ứng dụng di động dựa trên các đặc trưng như giờ trong ngày, loại ứng dụng, hoặc tần suất sử dụng.

2. Học không giám sát (Unsupervised Learning):

- Mô hình được huấn luyện trên dữ liệu không gắn nhãn, nhằm phát hiện các mẫu ẩn hoặc cấu trúc dữ liệu.
- Ví dụ: Phân cụm người dùng thành các nhóm có hành vi sử dụng thiết bị tương tự nhau.

3. Học tăng cường (Reinforcement Learning)

- Mô hình học từ việc tương tác với môi trường và cải thiện hành vi của mình dựa trên phản hồi (reward) nhận được.
- Ví dụ: Tối ưu hóa giao diện ứng dụng dựa trên hành vi tương tác của người dùng theo thời gian.

Ứng dụng của máy học

Máy học đã và đang được ứng dụng rộng rãi trong nhiều lĩnh vực, từ thương mại điện tử, chăm sóc sức khỏe đến tài chính và giáo dục. Đặc biệt, trong lĩnh vực phân tích hành

vi người dùng thiết bị di động, máy học đóng vai trò quan trọng trong việc khai thác và phân tích dữ liệu lớn để hiểu rõ hơn về thói quen và nhu cầu của người dùng. Một số ứng dụng phổ biến bao gồm:

- Dự đoán sở thích và cá nhân hóa trải nghiệm người dùng.
- Phân cụm người dùng dựa trên hành vi sử dụng thiết bị.
- Phát hiện các mẫu hành vi bất thường, giúp nâng cao bảo mật và tối ưu hóa dịch vụ.

3.2 Mô hình K-Nearest Neighbors

K-Nearest Neighbors (KNN) là một trong những thuật toán học máy cơ bản và phổ biến nhất, thuộc nhóm học có giám sát. Thuật toán này sử dụng khoảng cách để xác định "sự gần gũi" giữa các điểm dữ liệu và đưa ra dự đoán cho các mẫu mới dựa trên các điểm lân cận trong tập dữ liệu huấn luyện. **Đối với bài toán phân loại**, KNN dự đoán nhãn của một điểm mới bằng cách xem nhãn của k điểm lân cận gần nhất trong không gian đặc trưng. **Đối với bài toán hồi quy**, KNN dự đoán giá trị bằng cách lấy trung bình (hoặc giá trị khác) của k điểm lân cận gần nhất.

3.2.1 Nguyên lý hoạt động

KNN không có giai đoạn huấn luyện thực sự. Thuật toán chỉ lưu trữ toàn bộ tập dữ liệu huấn luyện.

Ở giai đoạn dự đoán, mô hình sẽ tiến hành các bước:

- Tính toán khoảng cách giữa điểm dữ liệu mới với tất cả các điểm dữ liệu trong tập huấn luyện.
- Sắp xếp các điểm dữ liệu huấn luyện theo thứ tự khoảng cách từ gần nhất đến xa nhất.
- Lấy k điểm gần nhất.
- Với bài toán phân loại: Nhãn được dự đoán là nhãn phổ biến nhất (đa số) trong k điểm lân cận.

3.2.2 Khoảng cách sử dụng trong KNN

KNN dựa vào việc tính toán khoảng cách giữa các điểm dữ liệu, trong đó các loại khoảng cách phổ biến là:

Khoảng cách Euclidean:

$$d(x, y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$$

Đây là loại khoảng cách phổ biến nhất trong không gian đặc trưng.

Khoảng cách Manhattan:

$$d(x, y) = \sum_{i=1}^{n} |x_i - y_i|$$

Khoảng cách Minkowski (Minkowski Distance)**:

$$d(x, y) = \left(\sum_{i=1}^{n} |x_i - y_i|^p\right)^{1/p}$$

Với $p = 2$, đây là khoảng cách Euclidean; với $p = 1$, đây là khoảng cách Manhattan.

Khoảng cách Cosine (Cosine Similarity): Thích hợp với dữ liệu dạng vectơ, tính bằng:

$$d(x, y) = 1 - \frac{\sum_{i=1}^{n} x_i y_i}{\sqrt{\sum_{i=1}^{n} x_i^2} \cdot \sqrt{\sum_{i=1}^{n} y_i^2}}$$

3.2.3 Các siêu tham số quan trọng

Số lượng neighbor (k):

- Giá trị k quyết định số điểm gần nhất sẽ được sử dụng để đưa ra dự đoán.
- Nếu k quá nhỏ, mô hình dễ bị nhiễu bởi các điểm dữ liệu ngoại lai (overfitting).
- Nếu k quá lớn, mô hình có thể bỏ qua chi tiết của dữ liệu và không phản ánh tốt mối quan hệ cục bộ (underfitting).
- Lựa chọn k thường được tối ưu hóa bằng các phương pháp như cross-validation.

Loại khoảng cách (Distance Metric): Tùy thuộc vào bản chất của dữ liệu, việc chọn loại khoảng cách thích hợp rất quan trọng để tăng độ chính xác của KNN.

Trọng số của láng giềng (Weighting):

- Uniform (mặc định): Mỗi điểm láng giềng đóng góp như nhau vào quyết định.
- Distance-based: Điểm gần hơn sẽ có trọng số cao hơn.

3.3 Các phương pháp đánh giá hiệu suất

Đầu tiên ta định nghĩa tp (đúng thật), fp (đúng giả), tn (sai thật), và fn (sai giả) như bảng sau:

	Được dự đoán là đúng	**Được dự đoán là sai**
Thực tế là đúng	tp (đúng thật)	fp (đúng giả)
Thực tế là sai	fn (sai giả)	tn (sai thật)

- **Accuracy**: Tỉ lệ trường hợp dự đoán đúng thực tế (đúng thật và sai thật) so với tổng số mẫu dữ liệu.

$$\text{accuracy} = \frac{tp + tn}{\text{không gian mẫu}} \tag{1}$$

- **Precision**: Đặc trưng cho khả năng bộ phân loại không gán nhầm nhãn sai thành đúng. Precision bằng 1 khi không có trường hợp đúng giả.

$$\text{precision} = \frac{tp}{tp + fp} \tag{2}$$

- **Recall**: Đặc trưng cho khả năng bộ phân loại tìm thấy tất cả mẫu dữ liệu đúng là thực sự đúng. Recall bằng 1 khi không có trường hợp sai giả (tức không có trường hợp đúng bị gán nhãn nhầm thành sai).

$$\text{recall} = \frac{tp}{tp + fn} \tag{3}$$

- **Điểm F1**: Là trung bình điều hòa giữa precision và recall. Điểm F1 mang tính đại diện hơn trong việc đánh giá độ chính xác trên đồng thời precision và recall.

$$\text{F1} = \frac{2}{\text{precision}^{-1} + \text{recall}^{-1}} \tag{4}$$

Nhận xét:

- Accuracy có công thức tường minh và dễ diễn giải, tuy nhiên accuracy đo lường trên tất cả các nhãn mà không quan tâm đến độ chính xác của từng nhãn.
- Điểm F1 đánh giá độ chính xác của dữ liệu trên từng nhãn. Điểm F1 được tính toán trên nhóm mẫu đúng thật, trong khi accuracy được tính toán trên nhóm mẫu đúng thật và sai thật.

4 Thống kê mô tả dữ liệu

Đầu tiên ta load các thư viện cần thiết cho việc phân tích:

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

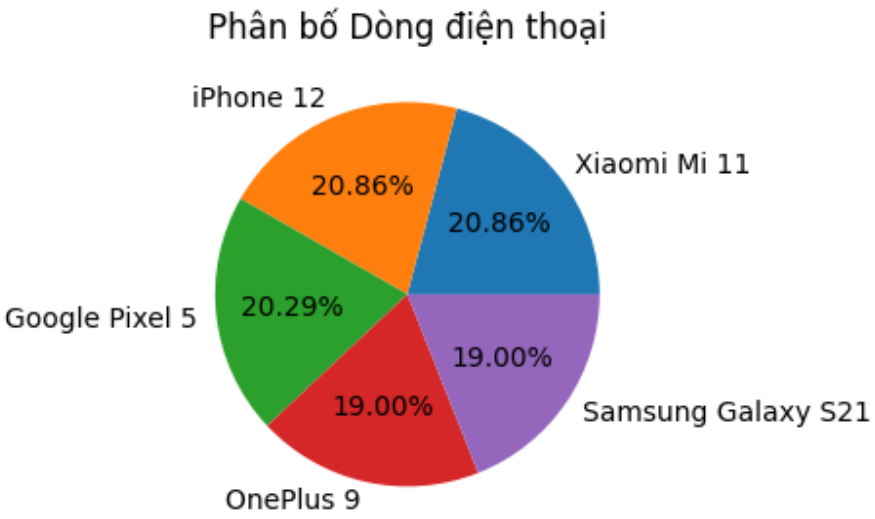
Đọc dữ liệu từ CSV vào biến df:

```
df = pd.read_csv(r"user_behavior_dataset.csv")
```

Đổi tên các trường và giá trị tiếng Anh sang tiếng Việt:

```
df.rename(columns={'Device Model': 'Dong thiet bi',
                   'Operating System': 'HDH',
                   'Gender': 'Gioi tinh',
                   'Age': 'Tuoi',
                   'Battery Drain (mAh/day)': 'Muc do hao pin (mAh/ngay)',
                   'Number of Apps Installed': 'So luong ung dung cai dat',
                   'Data Usage (MB/day)': 'Muc do su dung du lieu
                      (MB/ngay)',
                   'App Usage Time (min/day)': 'Thoi gian su dung ung dung
                      (phut/ngay)',
                   'Screen On Time (hours/day)': 'Thoi gian su dung man
                      hinh (gio/ngay)'},
                  inplace=True)

df['Gioi tinh'] = df['Gioi tinh'].replace({'Male': 'Nam', 'Female':
   'Nu'})
```

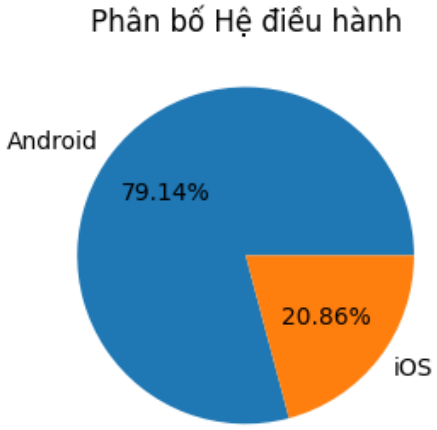
```
dev_count = df['Dong thiet bi'].value_counts()
plt.figure(figsize=(4, 3))
plt.pie(x=dev_count.values, labels=dev_count.index, autopct='%.2f%%')
plt.title("Phan bo Dong dien thoai")
plt.tight_layout()
plt.show()
```



Hình 1: Phân bố Dòng điện thoại

Nhận xét: Bộ dữ liệu này bao gồm 5 dòng điện thoại khác nhau. Mỗi dòng chiếm khoảng 20% của bộ dữ liệu.

```
os_count = df['HDH'].value_counts()
plt.figure(figsize=(4, 3))
plt.pie(x=os_count.values, labels=os_count.index, autopct='%.2f%%')
plt.title("Phan bo He dieu hanh")
plt.tight_layout()
plt.show()
```

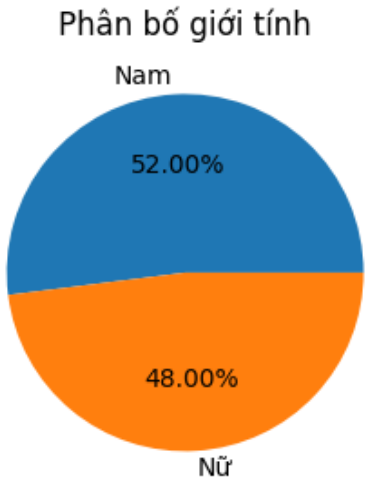


Hình 2: Phân bố Hệ điều hành

Nhận xét: Có hai loại hệ điều hành, Android chiếm khoảng 80% tập dữ liệu (điều này hợp lý vì 4 trong số 5 dòng điện thoại được khảo sát trong tập dữ liệu đều sử dụng

Android).

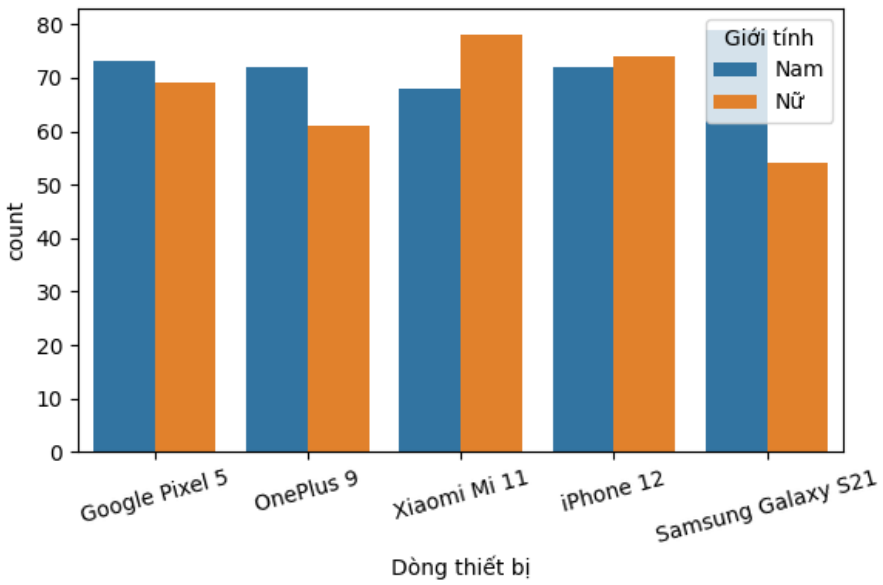
```
gender_count = df['Gioi tinh'].value_counts()
plt.figure(figsize=(4, 3))
plt.pie(x=gender_count.values, labels=gender_count.index,
    autopct='%.2f%%')
plt.title("Phan bo gioi tinh")
plt.tight_layout()
plt.show()
```



Hình 3: Phân bố Giới tính

Nhận xét: Bộ dữ liệu có tỷ lệ cân bằng giữa nam và nữ.

```
plt.figure(figsize=(6, 4))
ax = sns.countplot(data=df, x='Dong thiet bi', hue='Gioi tinh')
ax.tick_params(axis='x', labelrotation=15)
plt.tight_layout()
plt.show()
```

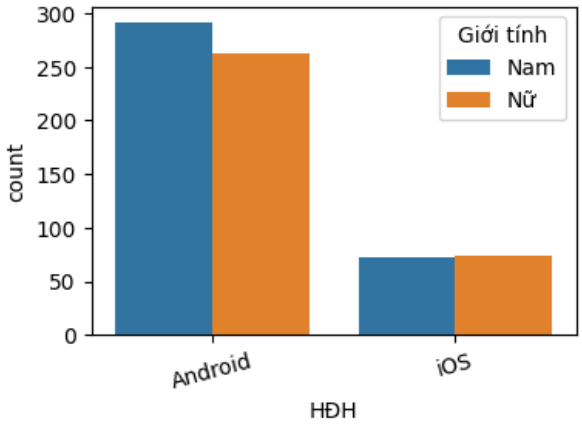


Hình 4: Phân bố tần số của Dòng thiết bị và Giới tính

Nhận xét: Biểu đồ trên giải thích mối quan hệ giữa dòng thiết bị và giới tính:

- Google Pixel 5, One Plus 9 và Samsung Galaxy 21 có nhiều người dùng nam hơn người dùng nữ và Xiaomi và iPhone 12 có nhiều người dùng nữ hơn người dùng nam.
- Mối quan hệ này không có lợi cho mục đích của bài tập lớn này, nhưng ta có thể mở rộng phạm vi nghiên cứu trong tương lai để chỉ ra một số mối quan hệ giữa biến mục tiêu (Lớp hành vi người dùng) và thương hiệu điện thoại, từ đó có thể cung cấp cho chúng ta thông tin liên quan đến giới tính và biến mục tiêu.

```
plt.figure(figsize=(4, 3))
ax = sns.countplot(data=df, x='HDH', hue='Gioi tinh')
ax.tick_params(axis='x', labelrotation=15)
plt.tight_layout()
plt.show()
```

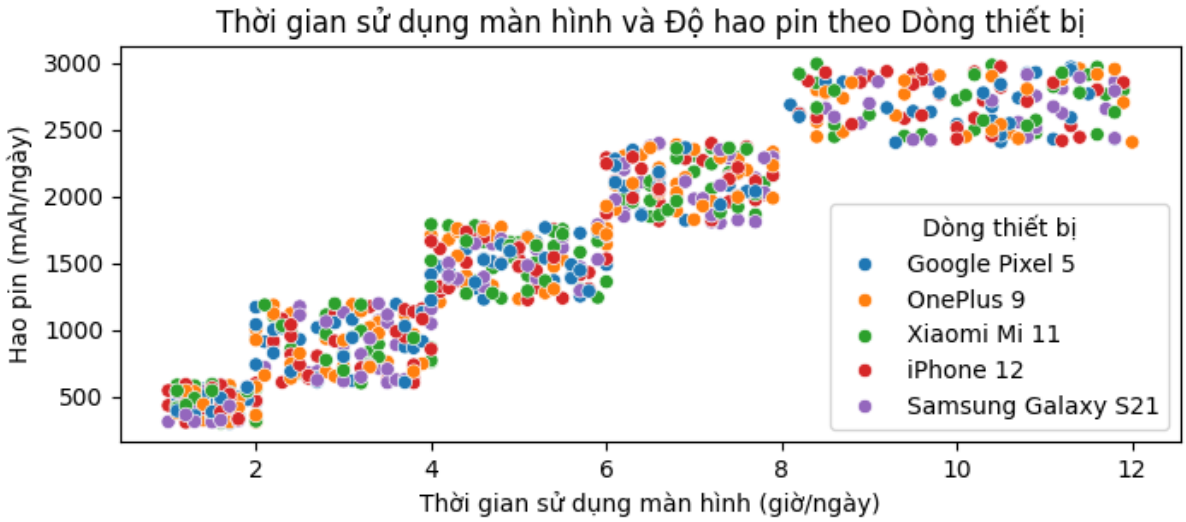


Hình 5: Phân bố tần số của Hệ điều hành và giới tính

Nhận xét: Biểu đồ trên tìm mối quan hệ giữa hệ điều hành và giới tính:

- Android có xu hướng có nhiều người dùng nam hơn, có thể là do các thiết bị cung cấp rộng hơn và tính linh hoạt.
- IOS cân bằng hơn về mặt phân bố giới tính, cho thấy nó hấp dẫn cả nam và nữ như nhau.

```
plt.figure(figsize=(8, 3))
ax = sns.scatterplot(x='Thoi gian su dung man hinh (gio/ngay)',
                     y='Muc do hao pin (mAh/ngay)',
                     hue='Dong thiet bi', data=df)
plt.title('Thoi gian su dung man hinh va Do hao pin theo Dong thiet bi')
plt.show()
```

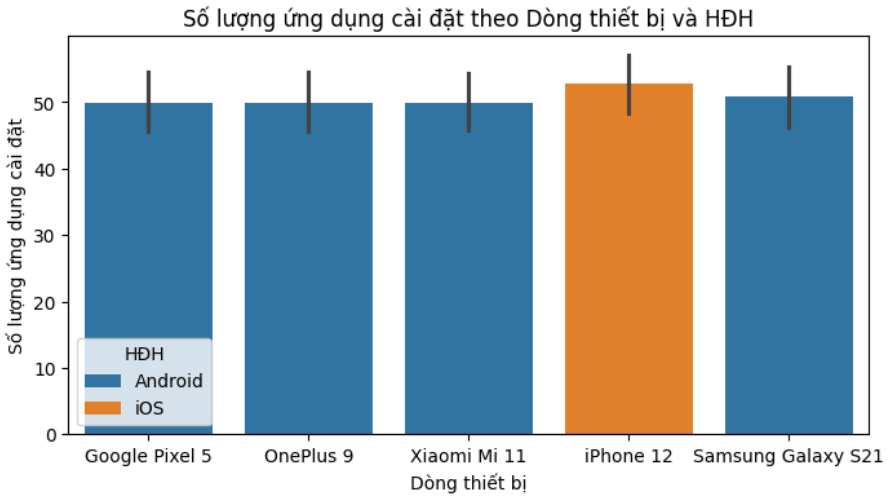


Hình 6: Thời gian sử dụng màn hình và Độ hao pin theo Dòng thiết bị

Nhận xét:

- Các nhóm phân biệt theo thời gian sử dụng màn hình: Các thiết bị thể hiện các mẫu hao pin, trong đó thời gian bật màn hình tập trung quanh các phạm vi cụ thể, chẳng hạn như 2, 4, 6, 8, 10 giờ/ngày.
- Hao pin tăng theo thời gian sử dụng màn hình: Như dự đoán, hao pin tăng theo thời gian bật màn hình dài hơn, với mức tiêu thụ mAh cao hơn khi thời gian sử dụng màn hình tăng lên.
- Không có thiết bị ngoại lệ đáng kể nào: Tất cả các dòng thiết bị - Google Pixel 5, OnePlus 9, Xiaomi Mi 11, iPhone 12 và Samsung Galaxy S21 - đều có phân bố tương tự trong các nhóm này, không có thiết bị nào nổi trội hơn hẳn về hiệu suất tốt hơn hay kém hơn.
- Phân tán lớn hơn ở thời gian sử dụng màn hình cao hơn: Ở thời gian sử dụng màn hình cao hơn (8-12 giờ), mức tiêu hao pin trên các thiết bị có sự thay đổi lớn hơn, cho thấy việc sử dụng kéo dài có thể có tác động khác nhau đến hiệu quả sử dụng pin.

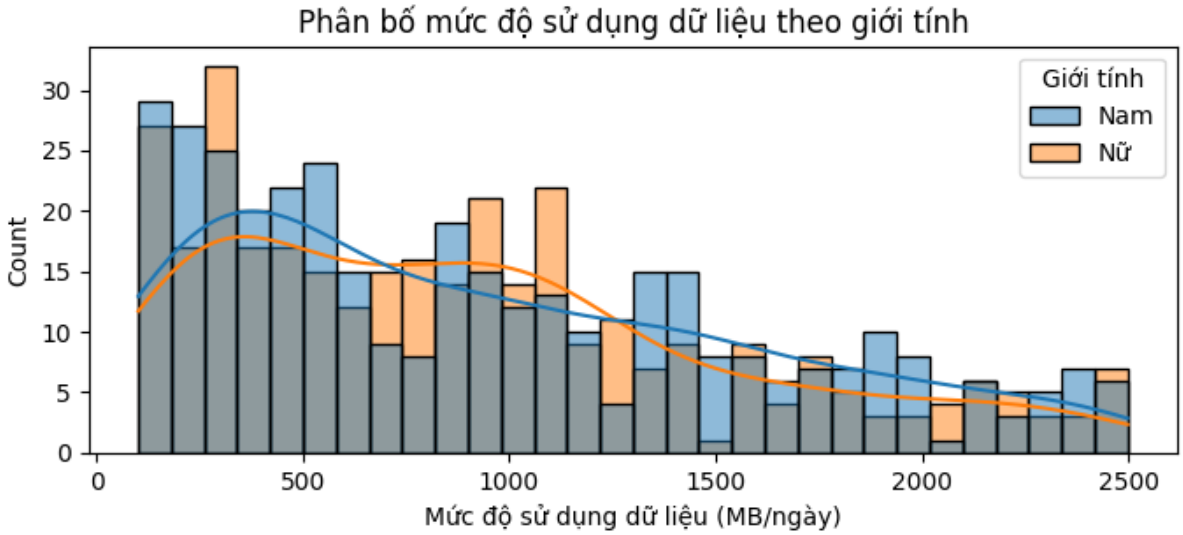
```
plt.figure(figsize=(8, 4))
sns.barplot(x='Dong thiet bi', y='So luong ung dung cai dat', hue='HDH',
    data=df)
plt.title('So luong ung dung cai dat theo Dong thiet bi va HDH')
plt.show()
```



Hình 7: Số lượng ứng dụng cài đặt theo Dòng thiết bị và Hệ điều hành

Nhận xét: Ứng dụng cài đặt phân bố đều theo các dòng thiết bị.

```
plt.figure(figsize=(8, 3))
ax = sns.histplot(data=df, x='Muc do su dung du lieu (MB/ngay)',
    hue='Gioi tinh', bins=30, kde=True)
plt.title('Phan bo muc do su dung du lieu theo gioi tinh')
plt.show()
```

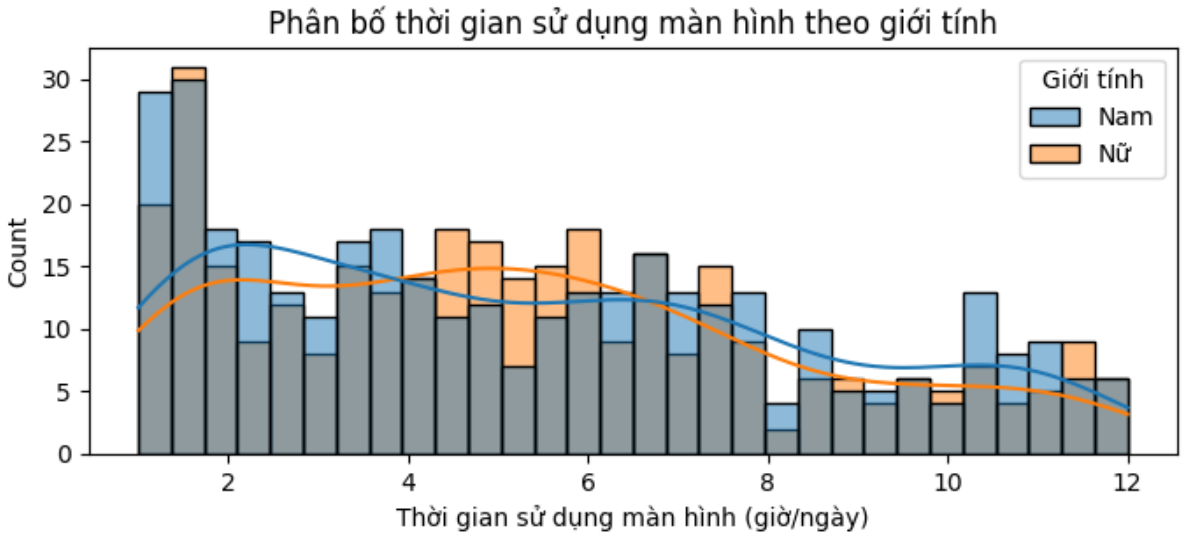


Hình 8: Phân bố mức độ sử dụng dữ liệu theo giới tính

Nhận xét:

- Mật độ xác suất cao ở mức độ sử dụng dữ liệu thấp: Cả nam và nữ đều có số lượng người dùng cao nhất trong phạm vi sử dụng dữ liệu thấp hơn (0-500 MB/ngày). Điều này cho thấy rằng hầu hết người dùng, bất kể giới tính, đều sử dụng lượng dữ liệu tương đối nhỏ mỗi ngày.
- Nam và nữ có các đỉnh khác nhau: Mật độ xác suất đối với nam đạt đỉnh ở khoảng 500 MB/ngày, trong khi đối với nữ là khoảng 700 MB/ngày, cho thấy rằng phụ nữ có xu hướng sử dụng nhiều dữ liệu hơn một chút so với nam giới.
- Hình dạng phân bố chung: Phân bố cho cả hai giới đều giảm khi lượng dữ liệu sử dụng tăng, với cả hai nhóm đều có ít người dùng sử dụng hơn 1500 MB/ngày. Điều này cho thấy rằng việc sử dụng dữ liệu nặng ít phổ biến hơn đối với cả hai giới.
- Lượng dữ liệu sử dụng cao hơn ở nam giới: Trong các danh mục sử dụng dữ liệu cao hơn (trên 1000 MB/ngày), số lượng người dùng nam thường vượt quá số lượng người dùng nữ, đặc biệt là trong phạm vi 1500–2500 MB/ngày.
- Nữ giới chiếm ưu thế trong việc sử dụng dữ liệu tầm trung: Nữ giới dường như chiếm ưu thế trong việc sử dụng dữ liệu tầm trung (500-1000 MB/ngày), cho thấy tỷ lệ người dùng dữ liệu ở mức trung bình cao hơn so với nam giới.

```
plt.figure(figsize=(8, 3))
sns.histplot(data=df,
             x='Thoi gian su dung man hinh (gio/ngay)',
             hue='Gioi tinh', bins=30, kde=True)
plt.title('Phan bo thoi gian su dung man hinh theo gioi tinh')
plt.show()
```



Hình 9: Phân bố thời gian sử dụng màn hình theo giới tính

Nhận xét:

- Hầu hết người tham gia khảo sát dành 1-3 giờ/ngày sử dụng màn hình.
- Nam giới thường có thời gian sử dụng màn hình cao hơn nữ giới, đặc biệt là trong khoảng 1-5 giờ.
- Thời gian sử dụng màn hình của nữ giới tập trung nhiều hơn ở mức 4-6 giờ, và có tần số thấp ở thời gian sử dụng màn hình cao (trên 7 giờ).
- Nam giới có thời gian sử dụng màn hình trải rộng hơn, trong khi thời gian sử dụng của nữ giới tập trung hơn.
- Thời gian sử dụng màn hình ở cả hai giới đều đạt đỉnh ở mức 1-2 giờ/ngày.

5 Phân loại hành vi người dùng thiết bị di động với KNN

5.1 Tiền xử lý dữ liệu

Đầu tiên ta load thư viện và đọc dữ liệu:

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt

df = pd.read_csv(r"user_behavior_dataset.csv")
```

Do cột dữ liệu 'User ID' không có ý nghĩa cho mô hình phân loại, ta tiến hành drop cột này đi. Đồng thời, ta cũng mã hóa các biến phân loại thành biến định lượng để có thể đưa vào mô hình KNN:

```
# Xoa cot du lieu khong mong muon
df.drop(columns='User ID', inplace=True, axis=1)

# Ma hoa bien phan loai
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
for col in df.columns[df.dtypes=='object']:
    df[col] = le.fit_transform(df[col])
```

Cuối cùng ta tiến hành chia tập dữ liệu thành hai tập con huấn luyện và tập con kiểm tra, trong đó nhóm chọn tỉ lệ dữ liệu cho huấn luyện là 75% và dữ liệu cho kiểm tra chiếm 25%.

```
from sklearn.model_selection import train_test_split

def standardize(x):
    x = np.array(x)
    return (x - x.mean(axis=0)) / x.std(axis=0)

def split_standardize(df, label):
    x = df.drop(columns=label)
    y = df[label]
    x_train, x_test, y_train, y_test = train_test_split(x, y,
        test_size=0.25, random_state=20)
```

```
    x_train = standardize(x_train)
    x_test = standardize(x_test)
    return x_train, x_test, y_train, y_test

x_train, x_test, y_train, y_test = split_standardize(df, label='User
    Behavior Class')
```

5.2 Huấn luyện mô hình KNN

Ta tiến hành load các thư viện phục vụ cho việc xây dựng, huấn luyện và đánh giá mô hình KNN:

```
from sklearn.model_selection import RandomizedSearchCV
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import classification_report, confusion_matrix
```

Ta khởi tạo mô hình KNN với các tham số mặc định như sau:

- `n_neighbors`: Số lượng neighbor gần nhất (Mặc định 5).
- `weights`: Trọng số của các neighbor (Mặc định: '`uniform`' - tất cả neighbor có cùng trọng số).
- `algorithm`: Thuật toán tìm kiếm láng giềng gần nhất (mặc định: '`auto`', thuật toán tự động lựa chọn phù hợp).
- `leaf_size`: Kích thước là trong cây tìm kiếm (Mặc định: 30).
- `p`: Hệ số trong khoảng cách Minkowski (Mặc định: 2 (khoảng cách Euclidean)).
- `metric`: Hàm đo khoảng cách (mặc định là '`minkowski`').

```
knn = KNeighborsClassifier()
```

Tiếp theo để tối ưu hóa mô hình này, ta có thể sử dụng công cụ `RandomizedSearchCV` của thư viện `scikit-learn`. `RandomizedSearchCV` là một công cụ được sử dụng để tối ưu hóa siêu tham số cho mô hình học máy. Công cụ này thực hiện việc tìm kiếm ngẫu nhiên trên một tập các siêu tham số do người dùng cung cấp thay vì thử tất cả các tổ hợp. Điều này giúp giảm thời gian và tài nguyên cần thiết, đặc biệt khi không gian siêu tham số rất lớn.

Ở đây, ta chọn siêu tham số cần tối ưu hóa là `n_neighbors` với các giá trị có thể của nó được lấy từ 2 đến 31. Sau đó, ta khởi tạo một đối tượng `RandomizedSearchCV` với mô

hình cần tối ưu hóa là mô hình đã khởi tạo ở trên, sử dụng độ chính xác (accuracy) làm tiêu chí để đánh giá mô hình, và sử dụng 10-fold cross-validation để đánh giá hiệu năng của mô hình.

10-fold cross-validation nghĩa rằng tập dữ liệu ban đầu được chia ngẫu nhiên thành 10 phần bằng nhau (hoặc gần bằng nhau) gọi là folds. Qua trình huấn luyện sẽ lặp qua 10 lần (mỗi lần là một fold), trong đó 1 fold được chọn làm tập kiểm tra và 9 fold còn lại được chọn làm tập huấn luyện. Công cụ `RandomizedSearchCV` sẽ huấn luyện mô hình trên tập huấn luyện và đánh giá hiệu suất trên tập kiểm tra. Sau khi lặp qua tất cả 10 folds, công cụ sẽ lấy trung bình các độ chính xác để đưa ra hiệu suất tổng thể của mô hình.

```
knn = KNeighborsClassifier()
params = {'n_neighbors': list(np.arange(2,32))}
model = RandomizedSearchCV(knn, random_state=20,
                           scoring='accuracy',
                           param_distributions=params,
                           cv=10)
```

Khi tiến hành huấn luyện, công cụ sẽ chọn ngẫu nhiên các giá trị của `n_neighbors` từ danh sách 2, 3, ..., 31 để thử nghiệm. Với mỗi giá trị được chọn, công cụ sẽ huấn luyện mô hình KNN. Sau cùng, `RandomizedSearchCV` sẽ trả về mô hình với siêu tham số tối ưu, tức là giá trị `n_neighbors` cho hiệu năng cao nhất (đo lường bằng accuracy).

Kết quả tối ưu của mô hình có thể được kiểm tra bằng property `model.best_params_` (giá trị tối ưu của `n_neighbors`) và `model.best_score_` (độ chính xác cao nhất đạt được với giá trị siêu tham số tối ưu).

```
def train(model, x_train, y_train):
    model.fit(x_train, y_train)
    print('Tham so toi uu: ', model.best_params_)
    print('Accuracy: ', model.best_score_)
    return model.best_estimator_

nknn = train(model, x_train, y_train)
```

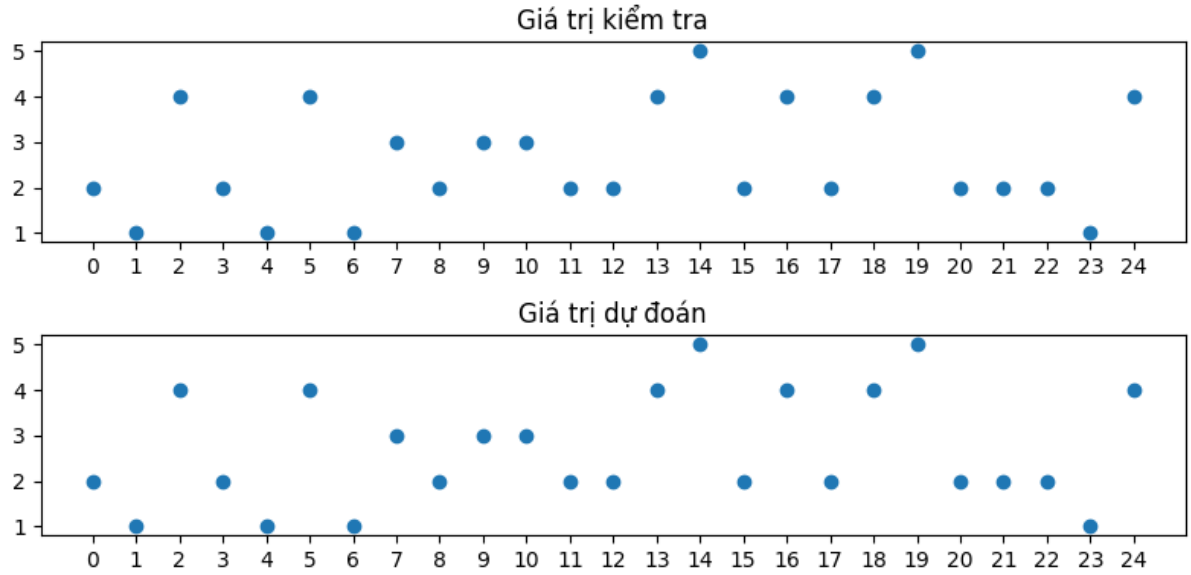
```
Tham so toi uu: {'n_neighbors': np.int64(3)}
Accuracy: 0.988534107402032
```

5.3 Dự đoán dựa trên mô hình đã huấn luyện

Ta sử dụng mô hình đã huấn luyện để dự đoán hành vi người dùng điện thoại dựa trên dữ liệu của tập kiểm tra:

```
def predict(trained_model, x_test):
    return trained_model.predict(x_test)

y_pred = predict(nknn, x_test)
```

Ta có thể vẽ mẫu một số điểm dữ liệu với đầu ra mong muốn và đầu ra dự đoán:

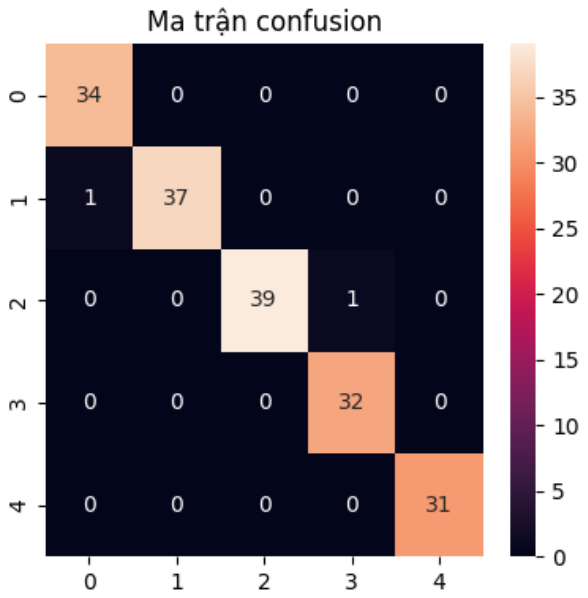


Hình 10: Vẽ mẫu một số điểm dữ liệu với đầu ra mong muốn và đầu ra dự đoán

Dựa trên đồ thị trên, ta có thể quan sát nhanh rằng 25 điểm dữ liệu đầu tiên đều được dự đoán chính xác.

Để đánh giá hiệu suất mô hình, ta có thể quan sát ma trận confusion:

```
def confusion_plot(y_test, y_pred):
    fig, ax = plt.subplots()
    fig.set_size_inches((4, 4))
    sns.heatmap(confusion_matrix(y_pred, y_test),
                annot=True, fmt='d', ax=ax)
    plt.title('Ma tran confusion')
    plt.tight_layout()
    plt.show()

confusion_plot(y_test, y_pred)
```



Từ kết quả trên, ta thấy rằng trong 175 điểm dữ liệu, chỉ có 2 điểm bị dự đoán sai.

```python
def report(y_test, y_pred):
    print(
        f"Bao cao hieu suat phan loai:\n"
        f"{classification_report(y_test, y_pred, digits=4)}\n"
    )

report(y_test, y_pred)
```

```
Bao cao hieu suat phan loai:
              precision   recall f1-score  support

           1     1.0000   0.9714   0.9855       35
           2     0.9737   1.0000   0.9867       37
           3     0.9750   1.0000   0.9873       39
           4     1.0000   0.9697   0.9846       33
           5     1.0000   1.0000   1.0000       31

    accuracy                        0.9886      175
   macro avg     0.9897   0.9882   0.9888      175
weighted avg     0.9889   0.9886   0.9886      175
```

Nhận xét: Phần lớn các điểm dữ liệu đều được dự đoán đúng.

- **Độ chính xác phân loại ở mỗi lớp**: Precision, recall và điểm F1 của tất cả các lớp đều dao động từ 0.98 đến 1.0. Lớp thứ 5 cho kết quả dự đoán chính xác nhất với điểm F1 là 1.0, trong khi lớp thứ 4 được dự đoán kém chính xác nhất với điểm

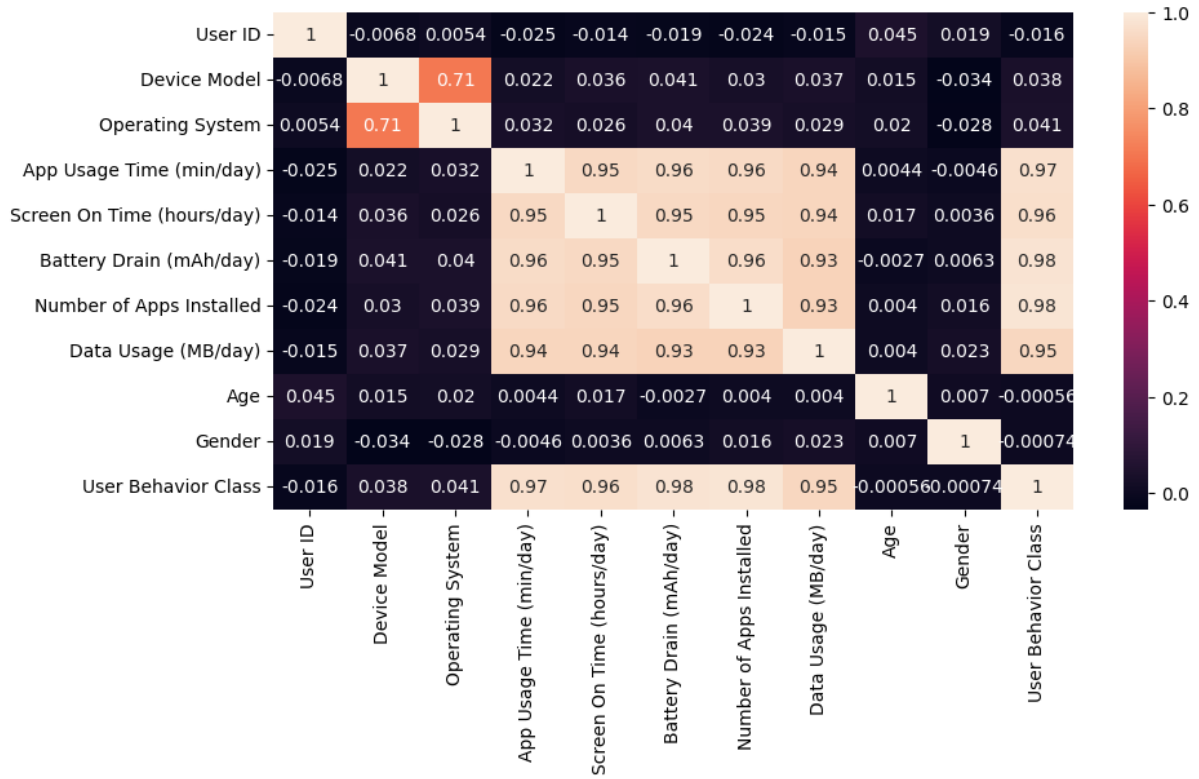
F1 là 0.9846.

- **Hiệu suất tổng thể của mô hình**: Mô hình có precision, recall, và điểm F1 trung bình đều cao là 0.9886.

5.4 Huấn luyện lại với các đặc trưng có tương quan lớn với ngõ ra

Ta tiến hành phân tích tương quan giữa các biến của dữ liệu ban đầu:

```
plt.figure(figsize=(10,5))
sns.heatmap(df.corr(), annot=True)
plt.show()
```



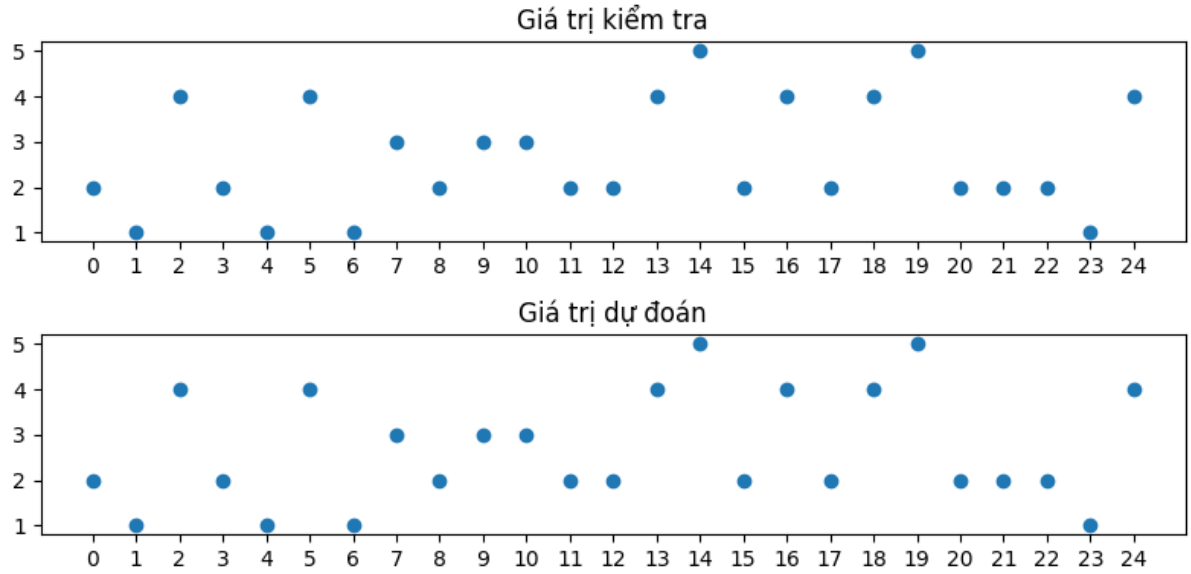
Hình 11: Ma trận tương quan giữa các biến của dữ liệu ban đầu

Quan sát kết quả trên, ta thấy rằng chỉ có các biến về Thời gian sử dụng ứng dụng, Thời gian sử dụng màn hình, Mức độ hao pin, Số lượng ứng dụng cài đặt, và Mức độ sử dụng dữ liệu là có tương quan rất cao so với biến ngõ ra. Trong khi đó, các biến còn lại có tương quan rất thấp với ngõ ra. Do đó, nhóm quyết định huấn luyện lại mô hình chỉ với các biến có tương quan cao với ngõ ra, và loại bỏ các biến có tương quan thấp khỏi dữ liệu:

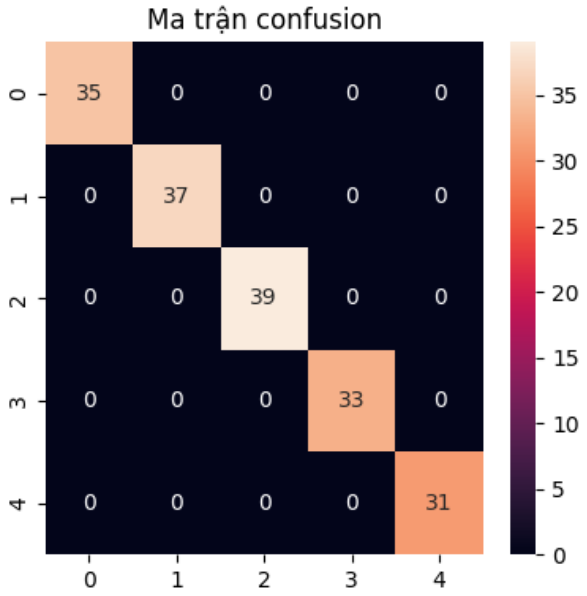
```
# Chon cac dac trung co tuong quan cao voi ngo ra
features = [
    'App Usage Time (min/day)',
    'Screen On Time (hours/day)',
    'Battery Drain (mAh/day)',
    'Number of Apps Installed',
    'Data Usage (MB/day)'
]
label = ['User Behavior Class']
df2 = df[features + label]

# Chia tap du lieu va chuan hoa
x_train2, x_test2, y_train2, y_test2 = split_standardize(df2,
    label='User Behavior Class')

# Huan luyen mo hinh
nknn2 = train(model, x_train2, y_train2)

# Du doan
y_pred2 = predict(nknn2, x_test2)

# Ve mau cac gia tri kiem tra va du doan
test_plot(y_test2, y_pred2)

# Ma tran confusion
confusion_plot(y_test2, y_pred2)

# Bao cao hieu suat
report(y_test2, y_pred2)
```

```
Tham so toi uu: {'n_neighbors': np.int64(15)}
Accuracy: 1.0
```

Ở mô hình này, số lượng neighbor tối ưu đã tăng từ 5 lên 15, và độ chính xác đã tăng đến 100%.



Vẽ mẫu một số điểm dữ liệu với đầu ra mong muốn và đầu ra dự đoán, ta có thể quan sát nhanh rằng 25 điểm dữ liệu đầu tiên đều được dự đoán chính xác.




```
Bao cao hieu suat phan loai:
              precision    recall  f1-score   support

           1     1.0000    1.0000    1.0000        35
           2     1.0000    1.0000    1.0000        37
           3     1.0000    1.0000    1.0000        39
           4     1.0000    1.0000    1.0000        33
           5     1.0000    1.0000    1.0000        31

    accuracy                         1.0000       175
```

```
   macro avg    1.0000    1.0000    1.0000       175
weighted avg    1.0000    1.0000    1.0000       175
```

Từ kết quả trên, ta thấy rằng tất cả các điểm trong 175 điểm dữ liệu đều được dự đoán đúng và điểm F1 cho tất cả các lớp cũng như cho tổng thể đều là 1.0.

6 Kết luận

Từ các kết quả đã phân tích và thử nghiệm, có thể thấy rằng mô hình KNN là một lựa chọn phù hợp và hiệu quả cho việc phân loại hành vi người dùng thiết bị di động. Qua việc sử dụng bộ dữ liệu chi tiết từ Kaggle, nhóm bài tập lớn đã đạt được độ chính xác cao khi áp dụng mô hình này, đặc biệt sau khi tối ưu hóa các siêu tham số và lựa chọn các đặc trưng có tương quan cao.

Các thí nghiệm đã chỉ ra rằng độ chính xác của mô hình KNN tăng đáng kể khi sử dụng quy trình tiền xử lý và tối ưu hóa siêu tham số thông qua công cụ `RandomizedSearchCV`. Việc huấn luyện lại mô hình chỉ với các đặc trưng có tương quan cao đã cải thiện kết quả, giúp mô hình đạt được độ chính xác 100%. Điều này cho thấy tầm quan trọng của việc lựa chọn và xử lý dữ liệu trong các ứng dụng học máy.

Tuy nhiên, nghiên cứu cũng nhận ra một số hạn chế của KNN, chẳng hạn như độ nhạy với dữ liệu không cân bằng hoặc chi phí tính toán cao với các tập dữ liệu lớn hơn. Trong tương lai, các giải pháp cải thiện có thể bao gồm thử nghiệm với các mô hình học máy khác như Random Forest hoặc SVM để so sánh hiệu suất, cũng như mở rộng phạm vi dữ liệu để kiểm chứng kết quả trên các tập dữ liệu lớn và đa dạng hơn.

Nhìn chung, nghiên cứu này không chỉ chứng minh tiềm năng của KNN trong phân loại hành vi người dùng mà còn mở ra nhiều cơ hội cho việc áp dụng máy học trong các lĩnh vực phân tích dữ liệu khác. Đây là một bước tiến quan trọng trong việc tận dụng dữ liệu để tối ưu hóa trải nghiệm người dùng và hỗ trợ phát triển các ứng dụng công nghệ thông minh.

References

[1] Christopher M Bishop and Nasser M Nasrabadi. *Pattern recognition and machine learning*. Vol. 4. 4. Springer, 2006.

[2] Alejandro G. Martín et al. "A survey for user behavior analysis based on machine learning techniques: current models and applications". In: *Applied Intelligence* 51.8 (2021), pp. 6029–6055.

[3] Joon-Myung Kang, Sin-seok Seo, and James Won-Ki Hong. "Usage pattern analysis of smartphones". In: *2011 13th Asia-Pacific Network Operations and Management Symposium*. IEEE. 2011, pp. 1–8.

[4] Andreas C Müller and Sarah Guido. *Introduction to machine learning with Python: a guide for data scientists*. " O'Reilly Media, Inc.", 2016.

[5] Foster Provost and Tom Fawcett. *Data Science for Business: What you need to know about data mining and data-analytic thinking*. " O'Reilly Media, Inc.", 2013.

[6] Jiqiang Song, Eugene Y Tang, and Leibo Liu. "User behavior pattern analysis and prediction based on mobile phone sensors". In: *Network and Parallel Computing: IFIP International Conference, NPC 2010, Zhengzhou, China, September 13-15, 2010. Proceedings*. Springer. 2010, pp. 177–189.