TRƯỜNG ĐẠI HỌC NGOẠI NGỮ - TIN HỌC THÀNH PHỐ HỒ CHÍ MINH
KHOA CÔNG NGHỆ THÔNG TIN

BÁO CÁO KẾT THÚC HỌC PHẦN

KHAI KHOÁNG DỮ LIỆU

Tìm hiểu các phương pháp kỹ thuật phân lớp có giám sát trên công cụ Orange

Giảng viên hướng dẫn: **ThS. Huỳnh Thành Lộc**

Sinh viên thực hiện:

1. Lê Phạm Hoàng Vũ 22DH114826
2. Uông Thành Trung 22DH113985
3. Trương Đình Trung 22DH113984

Thành phố Hồ Chí Minh, tháng 7 năm 2025

TRƯỜNG ĐẠI HỌC NGOẠI NGỮ - TIN HỌC THÀNH PHỐ HỒ CHÍ MINH
KHOA CÔNG NGHỆ THÔNG TIN

BÁO CÁO KẾT THÚC HỌC PHẦN

KHAI KHOÁNG DỮ LIỆU

Tìm hiểu các phương pháp kỹ thuật phân lớp có giám sát trên công cụ Orange

Mã lớp học phần: **241123018401**

Năm học: **2024 – 2025**

Học kỳ: **3**

Sinh viên thực hiện:

1. Lê Phạm Hoàng Vũ 22DH114826
2. Uông Thành Trung 22DH113985
3. Trương Đình Trung 22DH113984

Thành phố Hồ Chí Minh, tháng 7 năm 2025

LỜI CẢM ƠN

Trước hết, nhóm chúng em xin gửi lời cảm ơn sâu sắc và lòng biết ơn chân thành đến quý Thầy Cô khoa Công nghệ Thông tin – Trường Đại học Ngoại ngữ – Tin học TP.HCM, đặc biệt là Thầy Huỳnh Thành Lộc– người đã tận tâm hướng dẫn, hỗ trợ và tạo điều kiện thuận lợi cho nhóm trong suốt quá trình học tập và thực hiện đề tài.

Do còn hạn chế về kiến thức và kinh nghiệm, bài làm chắc chắn không thể tránh khỏi những thiếu sót cả về nội dung lẫn hình thức trình bày. Nhóm rất mong nhận được sự góp ý, nhận xét quý giá từ quý Thầy Cô để bài báo cáo được hoàn thiện hơn và đạt kết quả tốt nhất.

Nhóm xin chân thành cảm ơn!

MỤC LỤC

Lời cảm ơn **4**

DANH MỤC HÌNH ..... **i**

DANH MỤC BẢNG ..... **iii**

CHƯƠNG 1. GIỚI THIỆU ĐỀ TÀI **1**

1.1. Giới thiệu bài toán 1

1.2. Quá trình phân lớp dữ liệu 3

1.3. Tổng quan về phần mềm Orange 6

1.4. Các công trình liên quan 3

1.5. Một số phương pháp phân lớp dữ liệu sử dụng trong đề tài 4

CHƯƠNG 2. CHUẨN BỊ DỮ LIỆU **16**

2.1. Giới thiệu tập dữ liệu 16

2.2. Tiền xử lý và trực quan hóa dữ liệu 17

CHƯƠNG 3. Thực nghiệm mô hình trên orange **34**

3.1. Mô hình kNN (k Nearest Neighbor) 34

3.2. Mô hình Cây quyết định (Decision tree) 40

3.3. Mô hình Naive Bayes 49

3.4. Thước đo đánh giá mô hình 54

CHƯƠNG 4. KẾT QUẢ THỰC NGHIỆM **69**

4.1. *P*ipeline xử lý bài toán dự đoán sống sót trên tàu Titanic trên Orange 69

4.2. Kết quả dự đoán cuối cùng 71

CHƯƠNG 5. KẾT LUẬN **75**

5.1. Kết quả đạt được 75

5.2. Những khó khăn, hạn chế 76

TÀI LIỆU THAM KHẢO **78**

PHỤ LỤC **79**

DANH MỤC HÌNH

Hình 1.1. Ví dụ minh họa về phân loại 2
Hình 1.2 Quá trình phân lớp dữ liệu.....3
Hình 1.3 Ví dụ minh họa quá trình phân lớp huấn luyện.....4
Hình 1.4 Minh họa đánh giá mô hình.....5
Hình 1.5 Minh họa phân lớp dữ liệu mới 6
Hình 1.6 Tool Data 7
Hình 1.7 Tool Transform.....7
Hình 1.8 Tool Visualize 8
Hình 1.9 Tool model 1
Hình 1.10 Tool Evaluate 2
Hình 1.11 Tool Unsupervised 2
Hình 1.12 Minh họa công thức KNN 5
Hình 1.13 Minh họa KNN.....6
Hình 1.14 Minh họa ý tưởng Decision Tree.....7
Hình 1.15 Công Thức Entropy 10
Hình 1.16 Công thức Information Gain 11
Hình 1.17 Công Thúc GI.....12
Hình 2.1 Tool Data.....18
Hình 2.2 Kéo thả widget file 18
Hình 2.3 Cửa sổ widget.....19
Hình 2.4 Transfrom 20
Hình 2.5 Select columns.....20
Hình 2.6 Cấu hình 21
Hình 2.7 Select column in orange 22
Hình 2.8 Inpute.....23
Hình 2.9 Nối các widget.....24
Hình 2.10 Giao diện widget 25
Hình 2.11 Widget 26

Hình 2.12 Nối widget 26
Hình 2.13 Giao diện widget 27
Hình 2.14 widget 28
Hình 2.15 Nối widget 29
Hình 2.16 Data info 30
Hình 2.17 Nối data info 30
Hình 2.18 Data info 31
Hình 2.19 Data table 32
Hình 2.20 Data table 33
Hình 3.1 Widget 38
Hình 3.2 Nối widget 38
Hình 3.3 widget KNN 39
Hình 3.4 Xây dựng cây 46
Hình 3.5 Widget 47
Hình 3.6 Nối widget 47
Hình 3.7 Widget Decision tree 48
Hình 3.8 widget 52
Hình 3.9 Nối widget 53
Hình 3.10 Widget Naïve Bayes 54
Hình 3.11 Đánh giá test and Score 54
Hình 3.12 Đánh giá test and socre 55
Hình 3.13 Đánh giá so sánh 57
Hình 3.14 Đánh giá so sánh 59
Hình 3.15 Confusion Matrix KNN 61
Hình 3.16 Confusion Matrix Decision Tree 63
Hình 3.17 Confusion Matrix Naïve Bayes 65
Hình 4.1 Hình ảnh tổng quan về cách xử lý bài toán trong Orange 69
Hình 4.2 File (1) chứa dữ liệu dùng để test 70
Hình 4.3 Kết quả dự đoán 71

DANH MỤC BẢNG

Table 1 Mô tả thuộc tính và kiểu dữ liệu 16
Table 2 Dữ liệu mẫu 34
Table 3 Điểm cần dự đoán.....34
Table 4 Mô tả 35
Table 5 Mã hóa.....35
Table 6 Tính KNN.....36
Table 7 Dữ đoán 37
Table 8 Dữ liệu mẫu 41
Table 9 Điểm cần dự đoán.....41
Table 10 Mô tả ký hiệu.....43
Table 11 Mô tả sắp xếp 44
Table 12 IG của từng thuộc tính.....46
Table 13 Giải thích tham số 48
Table 14 Dữ liệu train.....50
Table 15 So sánh 52
Table 16 Tổng kết so sánh cả 3 mô hình.....67

CHƯƠNG 1. GIỚI THIỆU ĐỀ TÀI

1.1. Giới thiệu bài toán

Về khái niệm thì ta thấy phân lớp dữ liệu là một quá trình phân chia một đối tượng dữ liệu vào một hoặc nhiều lớp (loại) đã cho trước đó nhờ một mô hình phân lớp. Mô hình này được xây dựng dựa trên một tập dữ liệu đã được gán nhãn trước đó (thuộc về lớp nào). Quá trình gán nhãn thuộc lớp nào cho đối tượng trong dữ liệu chính là quá trình phân lớp dữ liệu.

Phân lớp dữ liệu là một kỹ thuật cần thiết trong việc thực hiện phân tích thông số, giải quyết các bài toán khó nhằn. Bên cạnh đó phương pháp này còn được ứng dụng thực tế rộng rãi trong nhiều ngành nghề khác nhau, chẳng hạn như trong tài chính ngân hàng, sales & marketing,… hay thậm chí là cả trong kinh tế học.

- Tài chính ngân hàng: giúp dự báo giá chứng khoán, xếp hạng tín dụng cá nhân và tổ chức, đánh giá rủi ro tài chính.
- Sales & marketing: dự báo được các khoản doanh thu cũng như số lượng khách hàng trung thành.
- Kinh tế học: nhằm giúp dự báo lượng cung cầu và khủng hoảng kinh tế.

Trong phạm vi đề tài này, nhóm tiến hành nghiên cứu và áp dụng các phương pháp phân lớp có giám sát (supervised learning) để giải quyết bài toán dự đoán khả năng sống sót của hành khách trên tàu Titanic, một thảm họa nổi tiếng trong lịch sử xảy ra vào năm 1912.

Tập dữ liệu được sử dụng là Titanic Dataset do Kaggle cung cấp, bao gồm thông tin chi tiết về các hành khách như: độ tuổi, giới tính, hạng vé, giá vé, số người thân đi cùng, cảng lên tàu, v.v…

Cột đích (Survived) là biến nhị phân với hai giá trị:

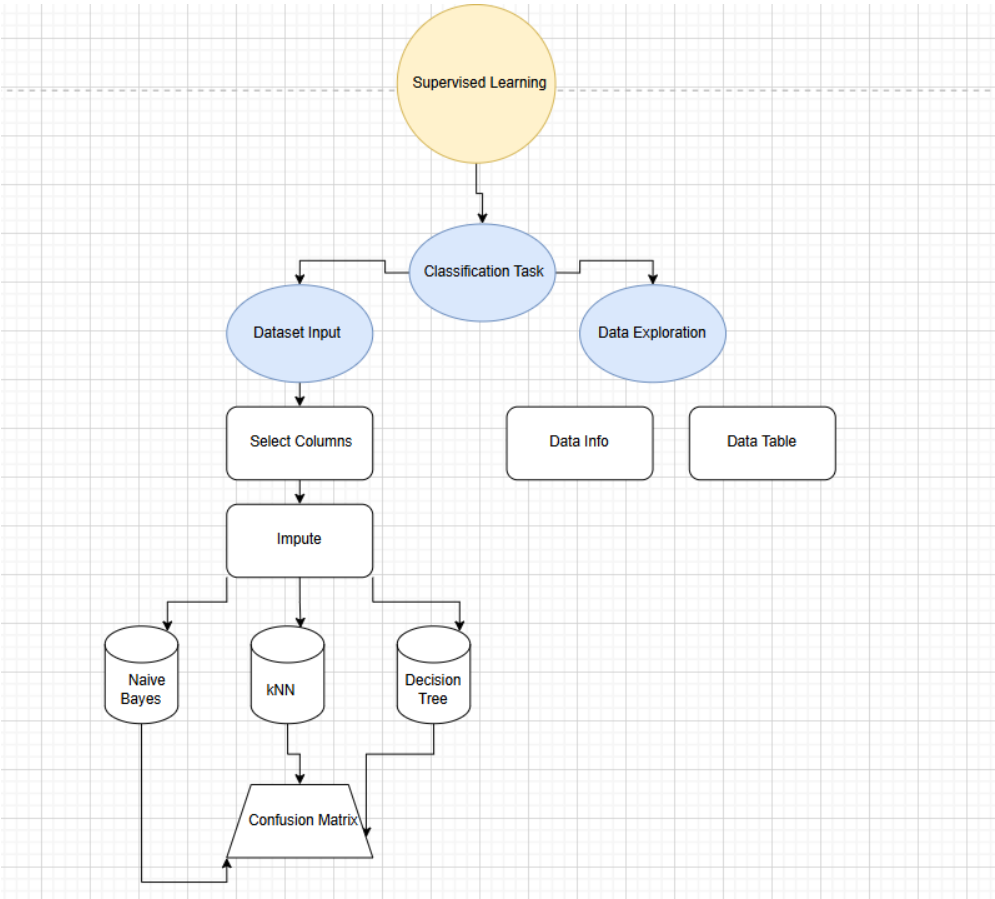
- 0: hành khách không sống sót
- 1: hành khách sống sót

Mục tiêu chính của bài toán là xây dựng mô hình có khả năng dự đoán chính xác khả năng sống sót của một hành khách mới dựa trên các thông tin cá nhân và điều kiện đi

tàu của họ. Đây là bài toán phân lớp nhị phân điển hình, cho phép đánh giá khả năng của các thuật toán học máy trong việc phân biệt và dự đoán nhãn đầu ra từ dữ liệu đặc trưng. Trong quá trình thực hiện, nhóm sử dụng phần mềm Orange Data Mining để:

- Tiền xử lý dữ liệu: làm sạch, xử lý dữ liệu thiếu, chuẩn hóa
- Trực quan hóa và hiểu rõ mối liên hệ giữa các đặc trưng
- Huấn luyện và đánh giá mô hình trên các thuật toán phân lớp tiêu biểu: k-Nearest Neighbors (kNN), Naive Bayes, và Decision Tree (Cây quyết định)

Bên cạnh việc đánh giá hiệu quả mô hình thông qua các chỉ số như độ chính xác (Accuracy), F1-score, độ nhạy (Recall), độ chính xác (Precision), MCC…, nhóm còn tập trung phân tích chi tiết nguyên lý hoạt động của từng thuật toán cũng như cách tính toán các chỉ số bằng tay, nhằm giúp người đọc nắm rõ bản chất của mỗi phương pháp.



Hình 1.1. Ví dụ minh họa về phân loại

1.2. Quá trình phân lớp dữ liệu

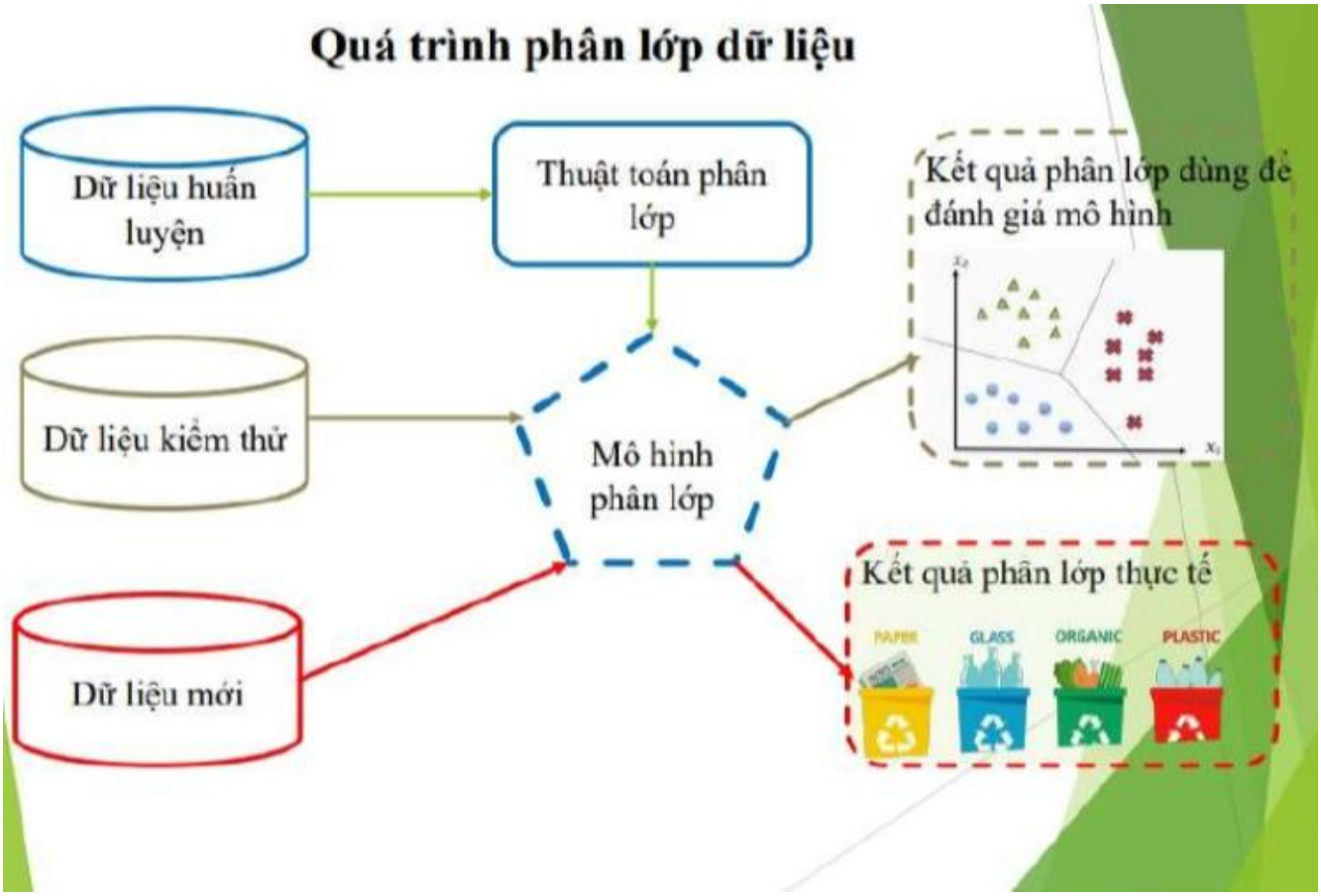
Quá trình phân lớp dữ liệu gồm 2 bước chính:

Bước 1: Xây dựng mô hình (giai đoạn "học" hoặc "huấn luyện")

Bước 2: Sử dụng mô hình chia thành 2 bước nhỏ

Bước 2.1: Đánh giá mô hình (kiểm tra tính đúng đắn của mô hình)

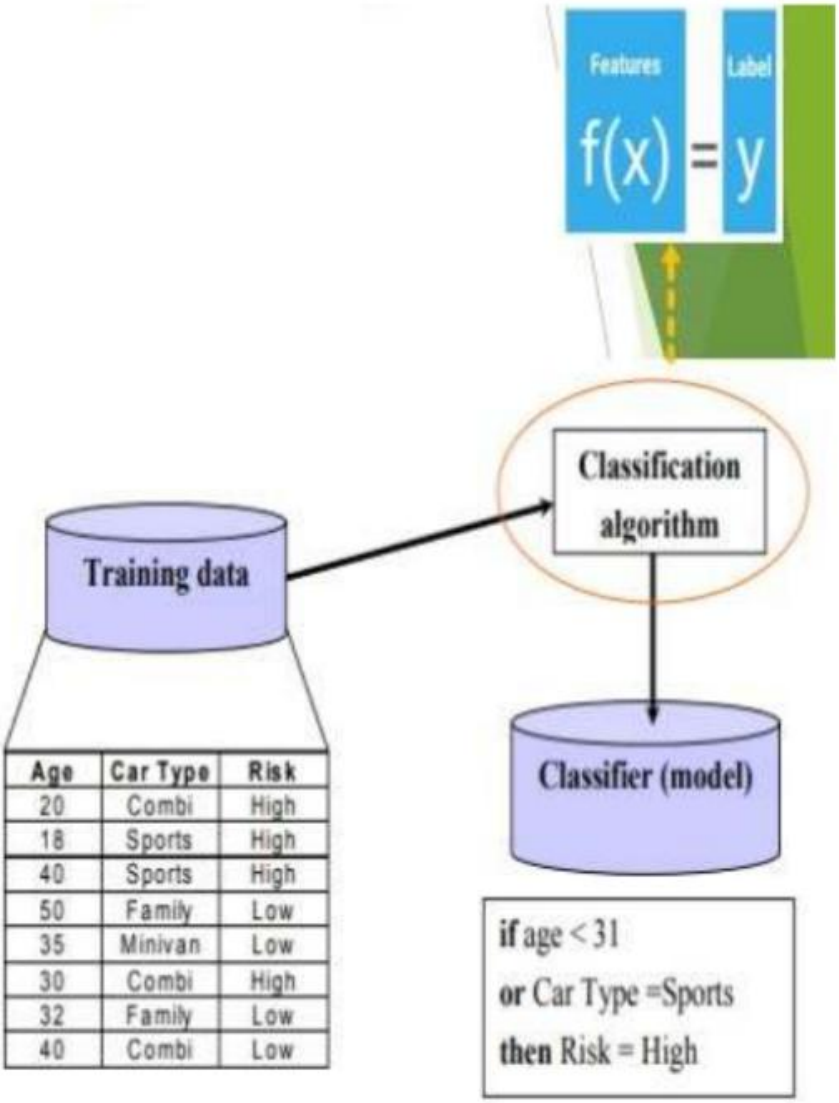
Bước 2.2: Phân lớp dữ liệu mới



Hình 1.2 Quá trình phân lớp dữ liệu

BƯỚC 1: Xây dựng mô hình phân lớp (huấn luyện) là quá trình huấn luyện hệ thống để phân loại dữ liệu đầu vào thành các nhóm nhất định. Đầu tiên, dữ liệu cần được thu thập, gán nhãn và tiền xử lý để đảm bảo độ chính xác. Sau đó, các thuật toán phân lớp như cây quyết định, hàm toán học hay tập luật được áp dụng để học quy tắc phân loại từ dữ liệu

huấn luyện. Kết quả là một mô hình (trình phân lớp) có thể dự đoán nhãn cho dữ liệu mới, hỗ trợ ra quyết định trong các lĩnh vực như tài chính, bảo hiểm và phân tích khách hàng.



Age	Car Type	Risk
20	Combi	High
18	Sports	High
40	Sports	High
50	Family	Low
35	Minivan	Low
30	Combi	High
32	Family	Low
40	Combi	Low

Hình 1.3 Ví dụ minh họa quá trình phân lớp huấn luyện

BƯỚC 2: SỬ DỤNG MÔ HÌNH CHIA THÀNH 2 BƯỚC NHỎ

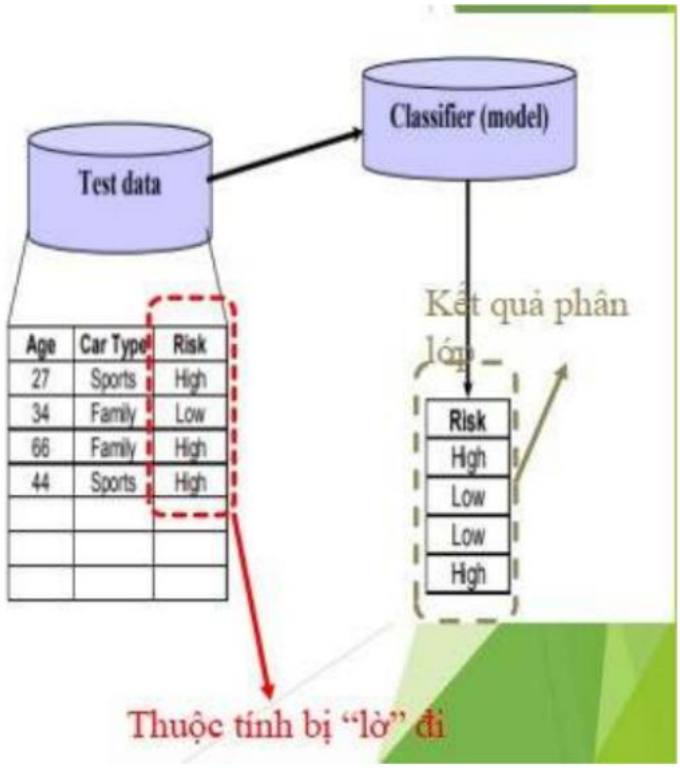
Bước 2.1: Đánh giá mô hình (kiểm tra tính đúng đắn của mô hình)

Đánh giá mô hình phân loại được thực hiện bằng cách sử dụng tập dữ liệu kiểm tra (Test data) – là dữ liệu khác với dữ liệu huấn luyện, đã được gán nhãn và xử lý trước. Khi

đánh giá, nhãn này được ẩn đi để kiểm tra khả năng học thực sự của mô hình. Mô hình sẽ dự đoán nhãn cho từng mẫu, sau đó so sánh với nhãn thực tế de do hieu suat.

Các chỉ số đánh giá phổ biến gồm:

- **Accuracy** (độ chính xác tổng thể),
- **Precision** (độ chính xác theo từng lớp),
- **Recall** (khả năng phát hiện đúng mẫu thuộc lớp),
- **F1-score** (cân bằng giữa Precision và Recall).

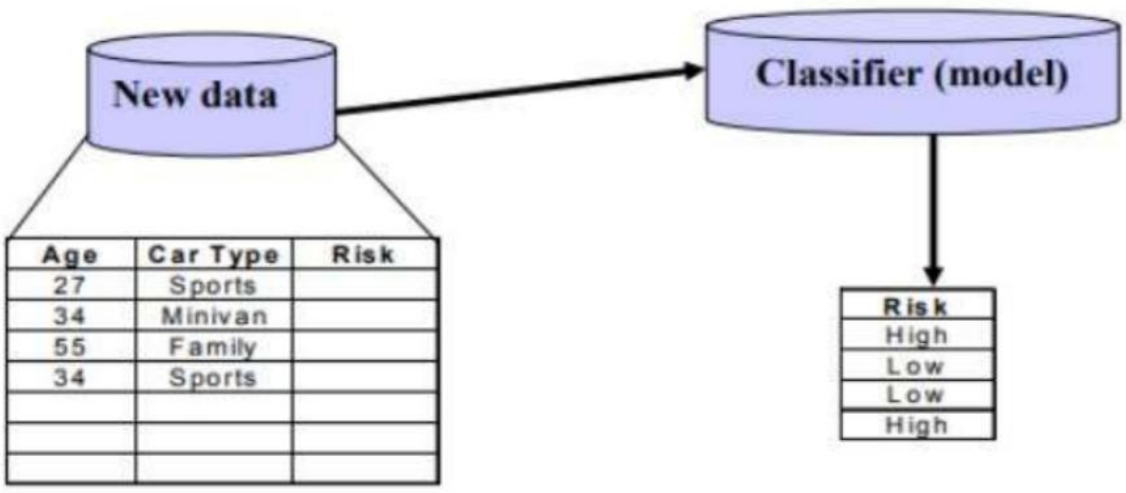


Hình 1.4 Minh họa đánh giá mô hình

Bước 2.2: Phân lớp dữ liệu mới

Trong giai đoạn phân lớp dữ liệu mới, mô hình học máy sẽ được áp dụng để dự đoán nhãn cho các mẫu dữ liệu chưa từng thấy trước đó. Dữ liệu đầu vào trong bước này là các đối tượngbị khuyết thuộc tính cần dự đoán lớp, tức là chưa có nhãn. Quá trình

này có ý nghĩa quan trọng trong thực tế, giúp tự động hóa việc phân loại thông tin trong nhiều lĩnh vực như tài chính, y tế, và hệ thống gợi ý. Hiệu suất của mô hình trong bước này sẽ phản ánh mức độ tổngquát hóa và khả năng áp dụng vào thực tiễn sau khi đã được huấn luyện ở bước 1.

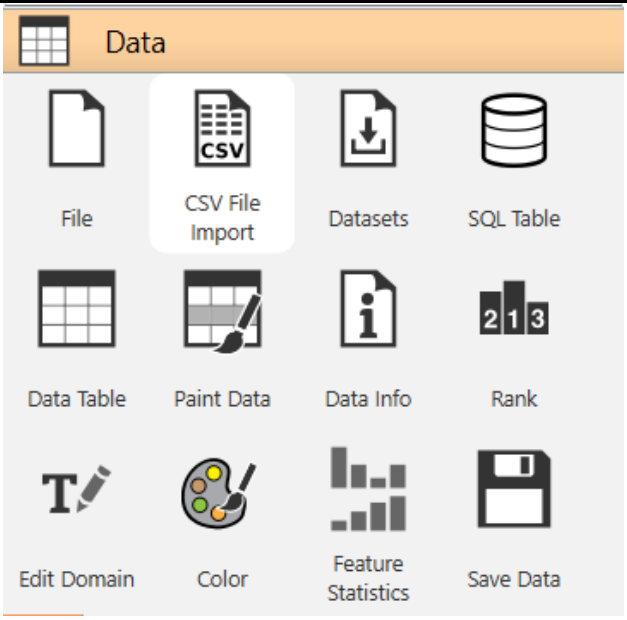


Hình 1.5 Minh họa phân lớp dữ liệu mới

1.3. Tổng quan về phần mềm Orange

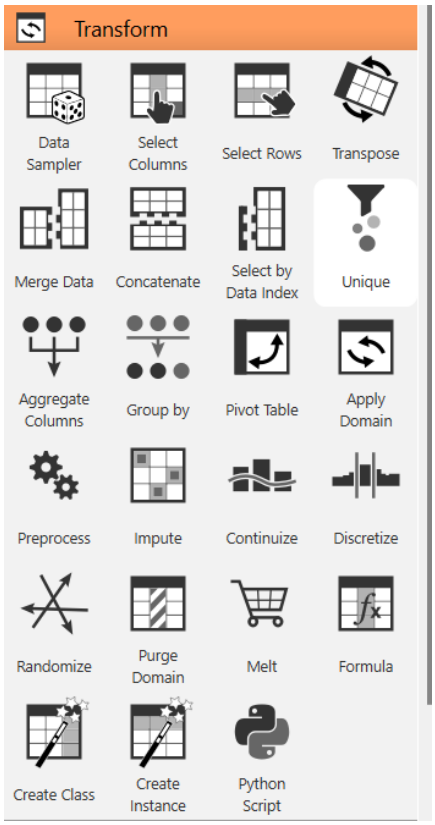
Phần mềm Orange là công cụ mã nguồn mở kết hợp khai thác dữ liệu và học máy, được viết bằng Python với giao diện trực quan. Orange phù hợp cho cả người mới lẫn người dùng chuyên nghiệp nhờ khả năng phân tích dữ liệu bằng thao tác kéo thả, không cần viết mã. Giao diện đồ họa sinh động, dễ sử dụng giúp nâng cao hiệu quả làm việc. Trên giao diện chính, các công cụ phân tích được sắp xếp thành 6 nhóm chức năng chính ở khung bên trái, hỗ trợ người dùng phân tích dữ liệu nhanh chóng và hiệu quả.

- Data: Data: rút trích, biến đổi và nạp dữ liệu (ETL process)



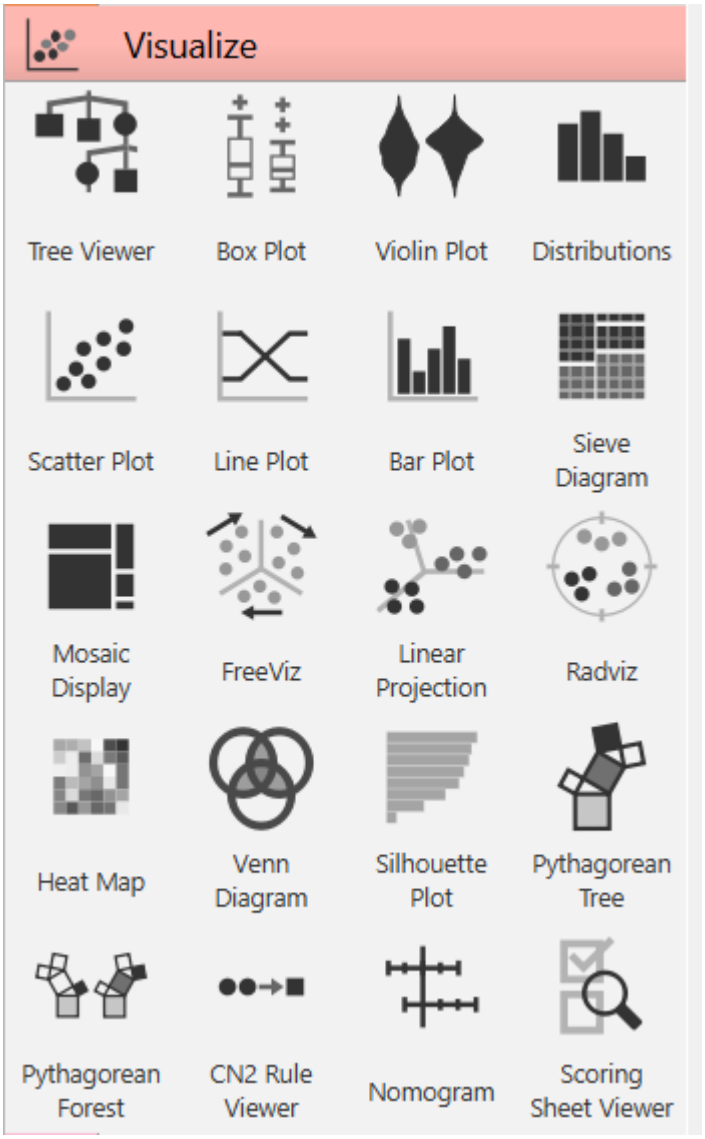
Hình 1.6 Tool Data

- Transform: hỗ trợ tiền xử lý và biến đổi dữ liệu để chuẩn bị cho phân tích



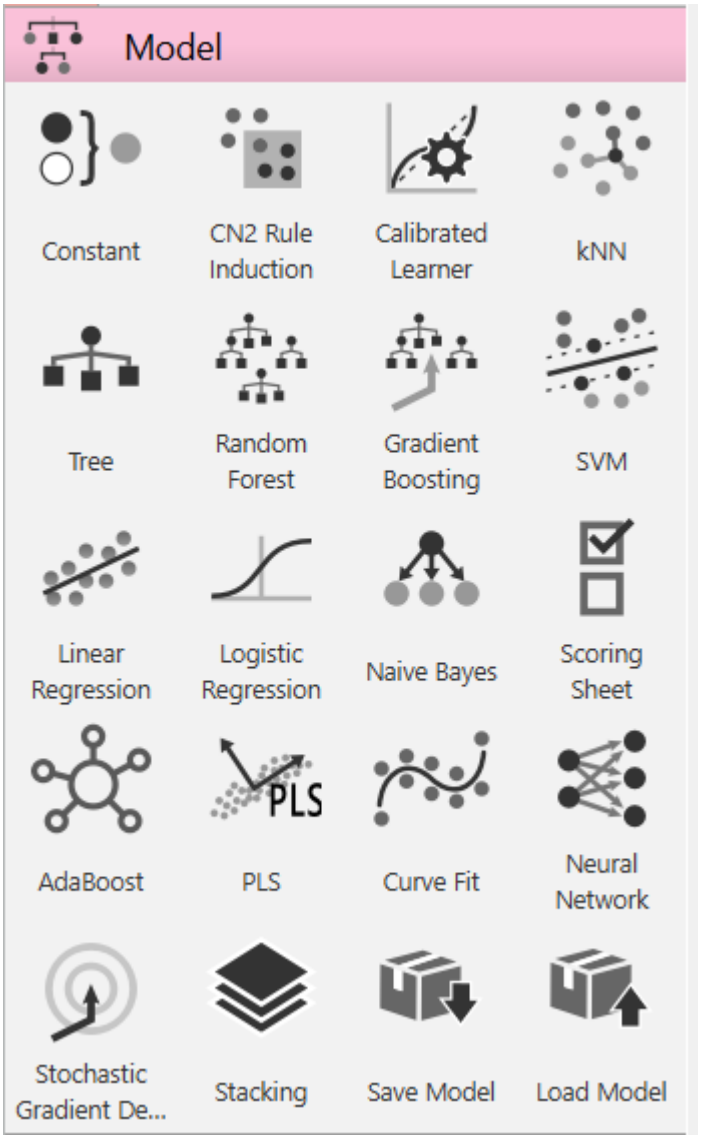
Hình 1.7 Tool Transform

- Visualize: biểu diễn biểu đồ, trực quan hóa dữ liệu giúp người dùng quan sát và hiểu dữ liệu tốt hơn.



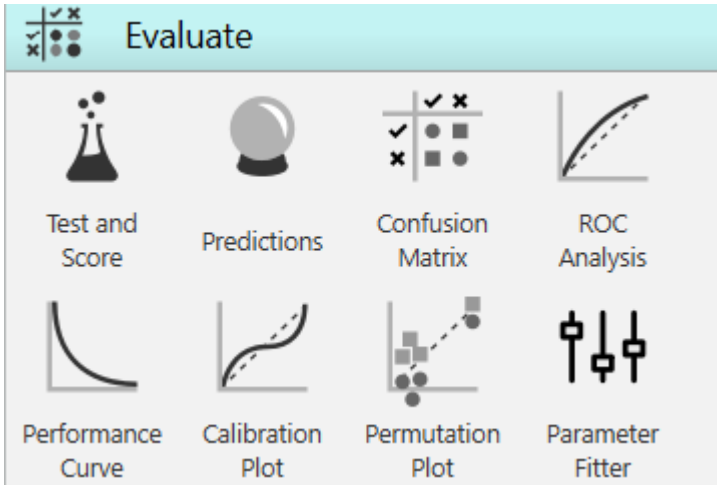
Hình 1.8 Tool Visualize

- Model: chứa các thuật toán học máy dùng để huấn luyện mô hình dự đoán như Tre, Logistic Regression, SVM,...



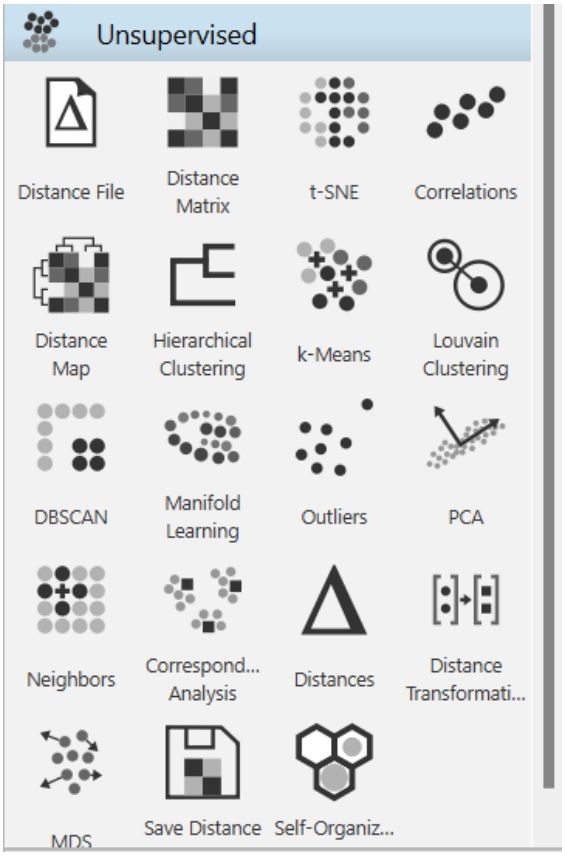
Hình 1.9 Tool model

- Evaluate: đánh giá hiệu suất của mô hình bằng các tiêu chí khác nhau như Test & Score, Confusion Matrix, ROC Analysis,..



Hình 1.10 Tool Evaluate

- Unsupervised: cung cấp các thuật toán phân cụm và giảm chiều dữ liệu như k-Mean, Hierachical Clustering,..



Hình 1.11 Tool Unsupervised

1.4. Các công trình liên quan

Trong lĩnh vực học máy, đặc biệt là bài toán phân lớp nhị phân có giám sát, nhiều mô hình đã được nghiên cứu và ứng dụng thành công. Các thuật toán phổ biến thường được áp dụng trong các bài toán như: phân loại thư rác, chẩn đoán bệnh, dự đoán khách hàng rời bỏ, và cả bài toán Titanic – dự đoán khả năng sống sót.

Một số mô hình nổi bật bao gồm:

- **Naive Bayes**: Là một thuật toán đơn giản nhưng mạnh mẽ dựa trên lý thuyết xác suất Bayes. Phù hợp với dữ liệu có phân phối chuẩn và tốc độ xử lý nhanh.
- **k-Nearest Neighbors (kNN)**: Là phương pháp không cần huấn luyện, phân loại mẫu mới dựa trên số đông trong k láng giềng gần nhất. Dễ hiểu, nhưng nhạy với nhiễu và kích thước dữ liệu lớn.
- **Decision Tree (Cây quyết định)**: Là mô hình trực quan hóa bằng cách phân nhánh dựa trên giá trị thuộc tính. Giúp người dùng dễ giải thích kết quả, nhưng dễ bị quá khớp nếu không cắt tỉa (prune).
- **Random Forest** (có thể đề cập thêm): Tập hợp nhiều cây quyết định để cải thiện độ chính xác, giảm overfitting.
- **Logistic Regression**: Là mô hình tuyến tính dùng phổ biến trong phân lớp nhị phân, nhất là trong thống kê và học máy cổ điển.
- **Neural Network (MLP – Multi-layer Perceptron):** Mạng nơ-ron nhân tạo gồm nhiều lớp: đầu vào – ẩn – đầu ra. Phù hợp với dữ liệu phi tuyến tính phức tạp
- **SVM (Support Vector Machine):** Tìm siêu phẳng tối ưu để phân chia dữ liệu thành các lớp.

1.5. Một số phương pháp phân lớp dữ liệu sử dụng trong đề tài.

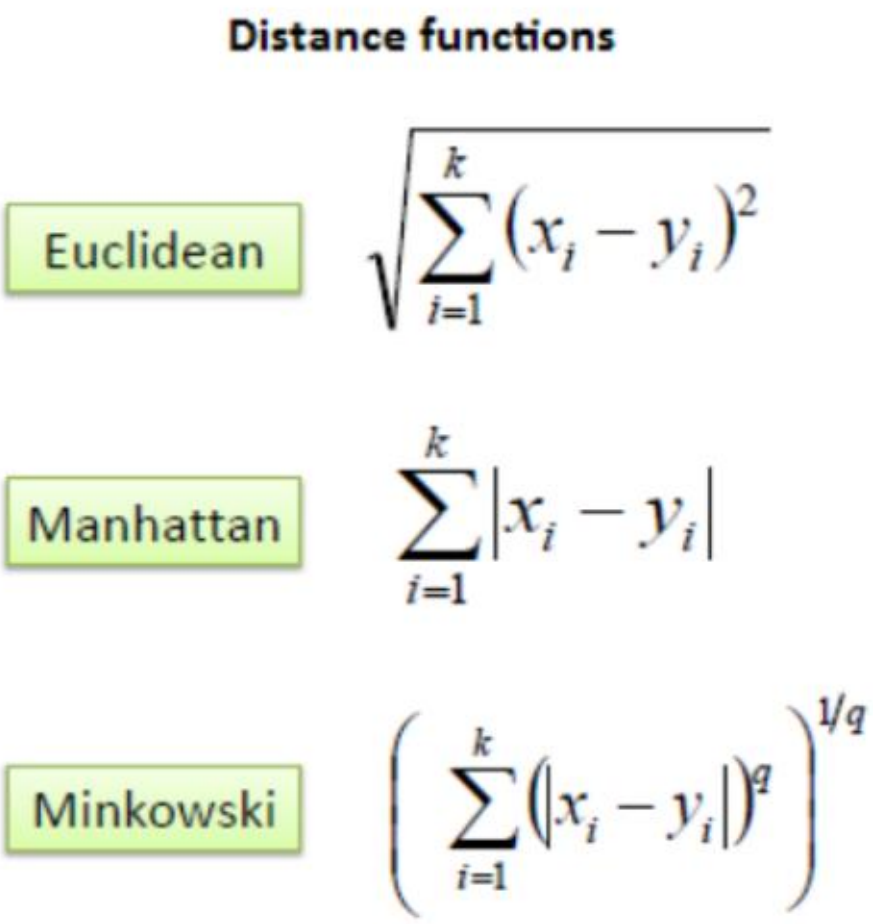
A. Phương pháp kNN (k Nearest Neighbor)

- KNN là một thuật toán học máy không giám sát, dùng cho phân loại và hồi quy. Thuật toán hoạt động bằng cách tìm k điểm gần nhất trong tập huấn luyện để dự đoán nhãn cho điểm mới. Không cần huấn luyện phức tạp, kNN chỉ lưu toàn bộ dữ liệu huấn luyện và so sánh khi cần.

- Ý tưởng chính: Dữ liệu tương tự thường nằm gần nhau trong không gian. Do đó, ta xác định k điểm gần nhất với điểm cần phân loại bằng cách tính khoảng cách (thường dùng các chuẩn như Euclidean, Manhattan hoặc Minkowski).

Tham số k ảnh hưởng lớn đến kết quả:

- **k quá nhỏ**: dễ bị nhiễu, sai lệch do dữ liệu bất thường.
- **k quá lớn**: mất tính "láng giềng gần nhất" vì trung bình hóa quá nhiều điểm.
- **k chẵn**: có thể gây khó phân loại do không rõ đa số.



Hình 1.12 Minh họa công thức KNN

Đặc điểm của KNN

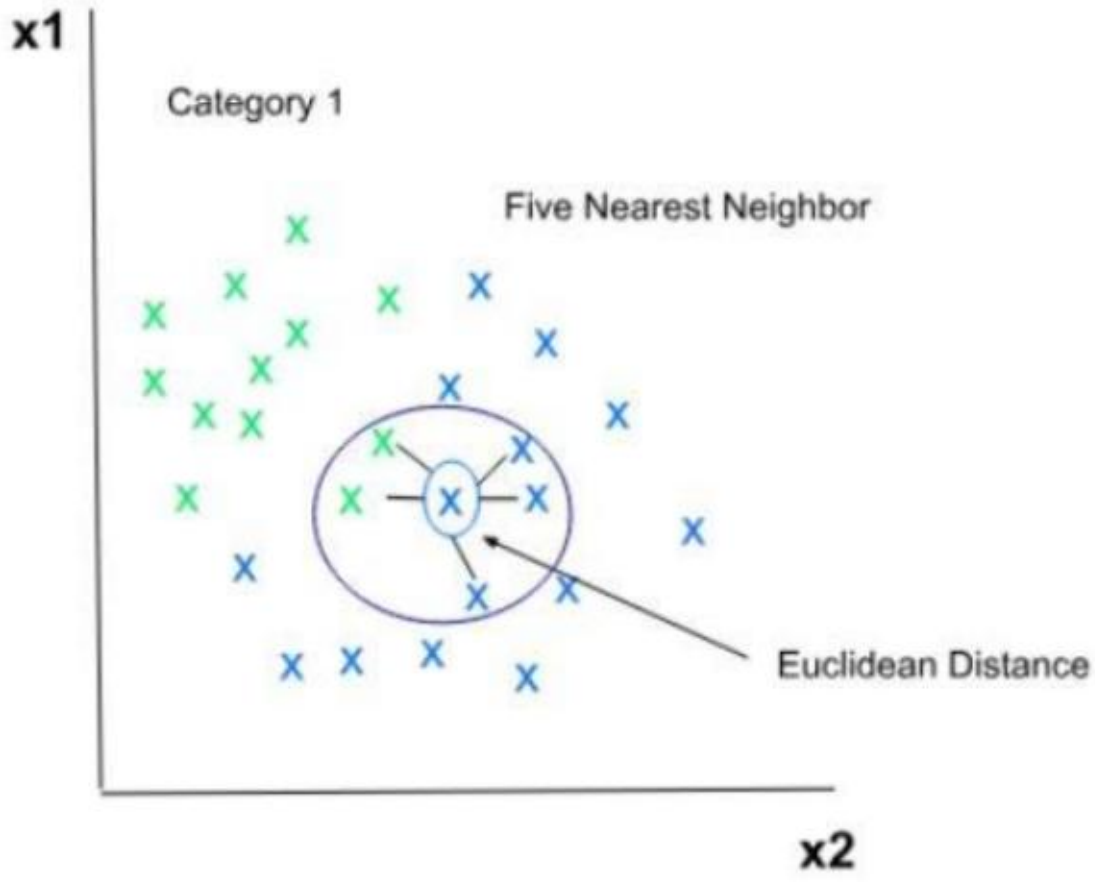
- Đơn giản, dễ hiểu, không yêu cầu giả định về phân phối dữ liệu.
- Không cần huấn luyện, chỉ lưu và so sánh dữ liệu khi cần dự đoán.
- Linh hoạt, áp dụng được cho nhiều bài toán và xử lý tốt phân loại nhiều lớp.
- Hiệu quả phụ thuộc vào cách chọn k và phương pháp tính khoảng cách (thường là Euclidean hoặc Manhattan).

Ưu điểm

- Dễ triển khai, không đòi hỏi kiến thức chuyên sâu.
- Làm việc tốt với dữ liệu không đồng nhất.
- Dễ điều chỉnh (qua k và cách đo khoảng cách).
- Cho kết quả chính xác nếu dữ liệu có cấu trúc rõ ràng.

Nhược điểm

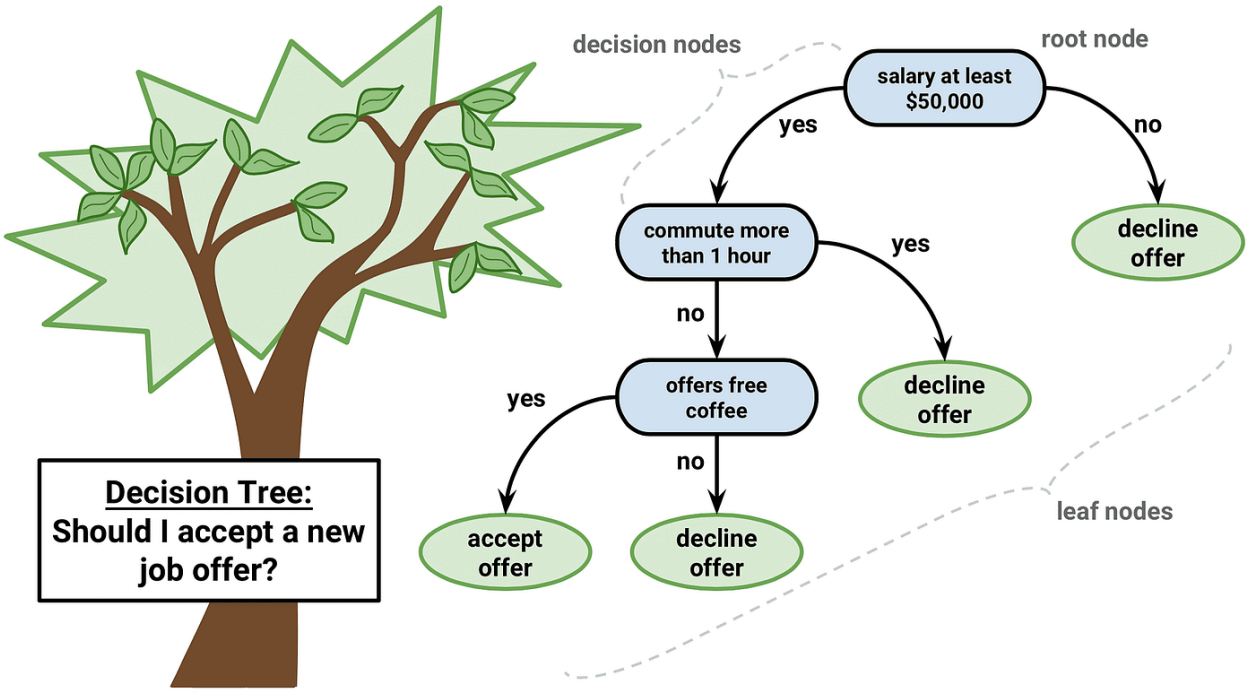
- **Tốn tài nguyên với tập dữ liệu lớn do phải tính khoảng cách với mọi điểm.**
- **Nhạy cảm với outliers.**
- **Việc chọn k không đơn giản, dễ gây overfitting hoặc underfitting.**
- **Không cung cấp độ tin cậy của dự đoán**



Hình 1.13 Minh họa KNN

B. Phương pháp Cây quyết định (Decision tree)

Cây quyết định là một mô hình dự đoán được sử dụng trong machine learning **và** khai phá dữ liệu, nhằm phân loại (classification) hoặc hồi quy (regression). Mô hình hoạt động bằng cách chia nhỏ tập dữ liệu thành các nhánh dựa trên các thuộc tính**,** sao cho các nhánh cuối (lá) đại diện cho kết quả dự đoán**.**



Hình 1.14 Minh họa ý tưởng Decision Tree

Ý tưởng:

- Tư duy giống con người: Cây quyết định mô phỏng quá trình ra quyết định của con người: nếu-then-else.
- Chia để trị (Divide and Conquer): Dữ liệu được chia nhỏ liên tục dựa trên giá trị thuộc tính giúp giảm độ không đồng nhất (impurity) của tập dữ liệu con.
- Tiêu chí chia: Tại mỗi nút, thuật toán chọn thuộc tính tốt nhất để phân chia dữ liệu, sao cho tập dữ liệu con càng "thuần nhất" càng tốt.

Đặc điểm:

- Mỗi nút trong cây là một thuộc tính.
- Mỗi nhánh là một giá trị của thuộc tính đó.
- Mỗi lá là một nhãn phân loại (trong classification) hoặc giá trị đầu ra (trong regression).
- Cây được xây dựng từ gốc đến lá bằng đệ quy.

Ưu điểm:

- **Dễ hiểu và trực quan**

Có thể biểu diễn dưới dạng cây nhánh rõ ràng, giống tư duy con người (if-then-else).

- Thích hợp để **trình bày, giảng dạy hoặc giải thích** với người không chuyên về kỹ thuật.
- **Không cần chuẩn hóa dữ liệu**

Không yêu cầu biến đầu vào phải có phân phối chuẩn hoặc chuẩn hóa đặc trưng như các thuật toán khác.

1. **Xử lý được cả dữ liệu rời rạc và liên tục**
   - Decision Tree có thể chia dữ liệu theo **giá trị phân loại** (ví dụ: Giới tính) hoặc **giá trị số** (ví dụ: Tuổi).
2. **Lựa chọn đặc trưng tự động**
   - Tự động chọn những thuộc tính có khả năng phân chia dữ liệu tốt nhất thông qua tiêu chí như Gini hay Entropy.
3. **Tốc độ huấn luyện nhanh**
   - So với nhiều thuật toán khác, cây quyết định có thời gian huấn luyện nhanh hơn (tùy theo số lượng thuộc tính và mức độ phân nhánh).

Nhược điểm:

1. **Dễ bị overfitting (quá khớp)**
    - Nếu không kiểm soát độ sâu hoặc không "cắt tỉa" (prune) cây, mô hình dễ ghi nhớ chi tiết dữ liệu huấn luyện → kém hiệu quả với dữ liệu mới.
2. **Không ổn định**
    - Chỉ cần thay đổi nhỏ trong dữ liệu cũng có thể **tạo ra cấu trúc cây hoàn toàn khác**.
3. **Không tối ưu trong mô hình hóa mối quan hệ phức tạp**
    - Đối với các bài toán có mối quan hệ phi tuyến tính phức tạp, Decision Tree có thể không đủ mạnh so với các mô hình như Random Forest hoặc Gradient Boosting.
4. **Thiên lệch nếu dữ liệu không cân bằng**
    - Nếu một lớp chiếm tỷ lệ quá lớn, cây sẽ ưu tiên lớp đó trong các nhánh → giảm hiệu suất của mô hình với lớp thiểu số.
5. **Không cung cấp xác suất tốt**
    - Kết quả đầu ra thường là một nhãn cụ thể, khó ước lượng chính xác xác suất cho mỗi lớp.

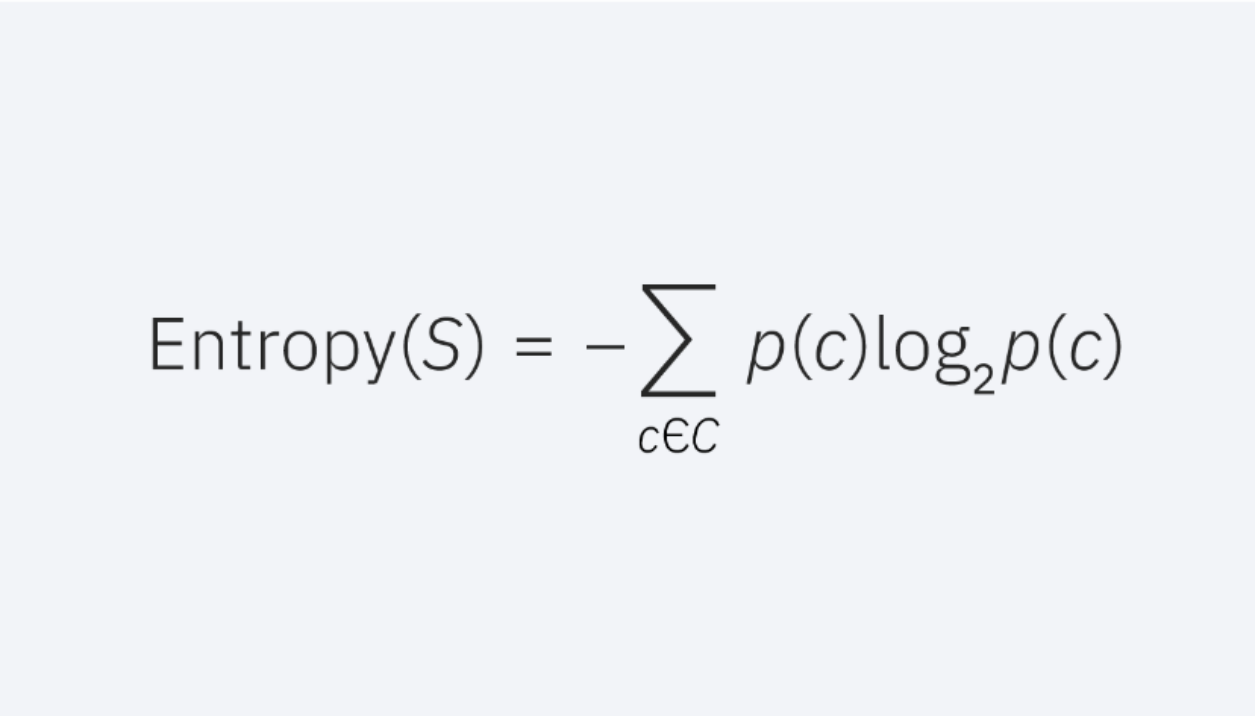
Các loại Decision Tree phổ biến:

- **ID3**: sử dụng **Entropy** và **Information Gain** làm tiêu chí chọn đặc trưng .
- **C4.5**: cải tiến từ ID3, dùng **Gain Ratio** để xử lý bias khi thuộc tính có nhiều giá trị .
- **CART** (Classification and Regression Tree): sử dụng tiêu chí **Gini Impurity** để đánh giá chất lượng phân chia .

Tiêu chí chọn thuộc tính tốt nhất (splitting criterion)

❖ Entropy và Information Gain

- **Entropy** đo độ hỗn loạn (impurity) trong một tập dữ liệu.
    - Giá trị Entropy càng thấp → dữ liệu càng thuần.



Hình 1.15 Công Thức Entropy

- S biểu thị tập dữ liệu mà entropy được tính toán
- c biểu thị các lớp trong tập hợp S
- p(c) biểu thị tỷ lệ các điểm dữ liệu thuộc lớp c so với tổng số điểm dữ liệu trong tập hợp S

- **Information Gain** = Entropy trước khi tách – trung bình Entropy sau tách.
    - Thuộc tính nào có **IG cao nhất** được chọn làm nút chia

$$\text{Information Gain}(S,a) = \text{Entropy}(S) - \sum_{v \in values(a)} \frac{|S_v|}{|S|} \text{Entropy}(S_v)$$

Hình 1.16 Công thức Information Gain

- a đại diện cho một thuộc tính cụ thể hoặc nhãn lớp.
  Entropy(S) là độ hỗn loạn (entropy) của toàn bộ tập dữ liệu S.
- |Sv| / |S| là tỷ lệ số mẫu trong tập con Sv so với tổng số mẫu trong tập S.
- |S|: Tổng số mẫu trong tập dữ liệu ban đầu.
- |Sv|: Số mẫu trong tập con Sv, tức là tập con của S ứng với giá trị v của thuộc tính a.

❖ **Gini Impurity**

- Đo xác suất một mẫu bị phân loại sai khi chọn ngẫu nhiên từ tập.
- Gini càng **thấp** → chất lượng chia càng tốt.

- CART chọn thuộc tính có **Gini impurity thấp nhất**

$$\text{Gini Impurity} = 1 - \sum_i (p_i)^2$$

Hình 1.17 Công Thúc GI

- **pi**: là **tỷ lệ mẫu thuộc lớp thứ iii** trong một tập dữ liệu (ví dụ: lớp 0, lớp 1,...)
- ∑i(pi)2: là **tổng bình phương xác suất** của từng lớp → thể hiện mức độ "thuần nhất" của tập dữ liệu.
- **Gini Impurity**: đo mức độ "lẫn lộn" giữa các lớp:

- Nếu tất cả mẫu thuộc **1 lớp duy nhất** → Gini = 0 (tập thuần nhất).
- Nếu mẫu phân bố đều giữa các lớp → Gini **cao hơn** (tập hỗn tạp).

C. Phương pháp Naïve Bayes

Naïve Bayes là thuật toán phân loại xác suất đơn giản (probabilistic classifier) dựa trên **Định lý Bayes**, giả định rằng **các đặc trưng (features) là độc lập có điều kiện**, tức là mỗi biến đầu vào không ảnh hưởng đến nhau nếu đã biết lớp mục tiêu

$$P(y|X) = \frac{P(X|y)P(y)}{P(X)}$$

- P(y|X) gọi là posterior probability: xác suất của mục tiêu y với điều kiện có đặc trưng X
- P(X|y) gọi là likelihood: xác suất của đặc trưng X khi đã biết mục tiêu y
- P(y) gọi là prior probability của mục tiêu y
- P(X) gọi là prior probability của đặc trưng X

Ở đây, **X** là vector các đặc trưng, có thể viết dưới dạng:

$$X = (x_1, x_2, x_3, ..., x_n)$$

Khi đó, đẳng thức Bayes trở thành:

$$P(y|x_1, ..., x_n) = \frac{P(x_1|y)P(x_2|y)...P(x_n|y)P(y)}{P(x_1)P(x_2)...P(x_n)}$$

Trong mô hình Naive Bayes, có hai giả thiết được đặt ra:

- Các đặc trưng đưa vào mô hình là độc lập với nhau. Tức là sự thay đổi giá trị của một đặc trưng không ảnh hưởng đến các đặc trưng còn lại.
- Các đặc trưng đưa vào mô hình có ảnh hưởng ngang nhau đối với đầu ra mục tiêu.

Khi đó, kết quả mục tiêu y để P(y|X) đạt cực đại trở thành:

$$y = argmax_y P(y) \prod_{i=1}^{n} P(x_i|y)$$

Chính vì hai giả thiết gần như không tồn tại trong thực tế trên, mô hình này mới được gọi là naive (ngây thơ). Tuy nhiên, chính sự đơn giản của nó với việc dự đoán rất nhanh kết quả đầu ra khiến nó được sử dụng rất nhiều trong thực tế trên những bộ dữ liệu lớn, đem lại kết quả khả quan.

❖ **Một số kiểu mô hình Naive Bayes**

- **Gaussian Naïve Bayes**: dùng với biến liên tục, giả định phân phối chuẩn cho từng lớp, ước tính thông qua trung bình và phương sai

$$P(x_i|y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} exp\left(-\frac{(x_i-\mu_y)^2}{2\sigma_y^2}\right)$$

- **Multinomial Naïve Bayes**: dùng với dữ liệu đếm (như từ trong văn bản), phù hợp cho mô hình phân loại văn bản

- **Bernoulli Naïve Bayes**: xử lý biến Boolean (0/1, True/False), phổ biến trong lọc spam

❖ **Ưu điểm**

- **Đơn giản, dễ hiểu và triển khai nhanh** nhờ mô hình không phức tạp và dùng bộ tham số rất nhỏ.
- **Hiệu quả với dữ liệu chiều cao (high-dimensional)** như phân loại văn bản, spam filtering.
- **Yêu cầu dữ liệu huấn luyện nhỏ** để ước lượng tham số, phù hợp khi bộ dữ liệu hạn chế.
- **Không cần điều chỉnh nhiều tham số**, dễ mở rộng để tạo mô hình mới nếu cần thường xuyên.

❖ **Nhược điểm**

- **Giả định độc lập có điều kiện không thực tế** trong phần lớn dữ liệu thực tế → có thể dẫn đến sai lệch trong xác suất dự đoán.
- **Zero-frequency problem**: nếu một giá trị thuộc tính không xuất hiện trong lớp nào đó → xác suất bằng 0 → kết quả cuối cùng cũng = 0. Cần dùng **Laplace hoặc Lidstone smoothing** để khắc phục.
- **Không phù hợp khi biến mang nhiều phụ thuộc** lẫn nhau, vì giả định độc lập bị vi phạm nghiêm trọng → hiệu quả giảm.
- **Xác suất hậu nghiệm có thể rất không chính xác**, dù nhãn dự đoán vẫn đúng nếu lớp đó có giá trị cao nhất.

❖ Ứng dụng thực tiễn

- **Lọc thư rác (spam filtering)**: sử dụng bag-of-words để phân loại email, phổ biến từ năm 1990s và vẫn rất thông dụng hiện nay.

- **Phân loại tài liệu và xử lý ngôn ngữ** (ví dụ: phân loại tin tức, sentiment analysis) với dữ liệu văn bản lớn và nhiều chiều.

CHƯƠNG 2. CHUẨN BỊ DỮ LIỆU

2.1. Giới thiệu tập dữ liệu

Trong đề tài này, nhóm sử dụng Titanic Dataset – một bộ dữ liệu nổi tiếng được dùng phổ biến trong các bài toán học máy cơ bản. Tập dữ liệu được công bố bởi Kaggle trong cuộc thi "Titanic: Machine Learning from Disaster", với mục tiêu dự đoán khả năng sống sót của hành khách trên chuyến tàu RMS Titanic – một thảm họa hàng hải xảy ra năm 1912, khiến hơn 1500 người thiệt mạng.

Đây là một bài toán phân lớp nhị phân có giám sát (supervised binary classification), trong đó biến đầu ra (target) là Survived – đại diện cho khả năng sống sót của hành khách.

Mục tiêu: Dựa vào các thuộc tính mô tả hành khách như tuổi, giới tính, giá vé, hạng ghế,... mô hình sẽ dự đoán hành khách đó có sống sót hay không khi xảy ra tai nạn.

Table 1 Mô tả thuộc tính và kiểu dữ liệu

Thuộc tính	Kiểu dữ liệu	Mô tả
PassengerId	Số nguyên	ID định danh từng hành khách (không dùng trong huấn luyện)
Survived	Nhị phân	Biến mục tiêu: 0 = không sống sót, 1 = sống sót
Pclass	Số nguyên	Hạng vé (1 = hạng 1, 2 = hạng 2, 3 = hạng 3) – phản ánh điều kiện kinh tế
Name	Chuỗi	Tên hành khách (thường bị loại bỏ do không mang giá trị phân lớp)
Sex	Chuỗi	Giới tính (nam/nữ)
Age	Số thực	Tuổi hành khách (có giá trị bị thiếu)
SibSp	Số nguyên	Số anh chị em/vợ/chồng đi cùng
Parch	Số nguyên	Số cha mẹ/con cái đi cùng
Ticket	Chuỗi	Mã số vé (không có giá trị phân loại rõ ràng, thường bị bỏ)

Fare	Số thực	Giá vé hành khách đã trả
Cabin	Chuỗi	Mã phòng (rất nhiều giá trị thiếu – thường bị loại)
Embarked	Chuỗi	Cảng nơi hành khách lên tàu (C, Q, S)

2.2. Tiền xử lý và trực quan hóa dữ liệu

Dữ liệu trong thực tế thường chứa nhiều vấn đề như giá trị thiếu, cột không cần thiết, dữ liệu nhiễu hoặc sai định dạng. Vì vậy, để đảm bảo mô hình học máy hoạt động hiệu quả, nhóm đã thực hiện quy trình tiền xử lý dữ liệu trước khi đưa vào huấn luyện.

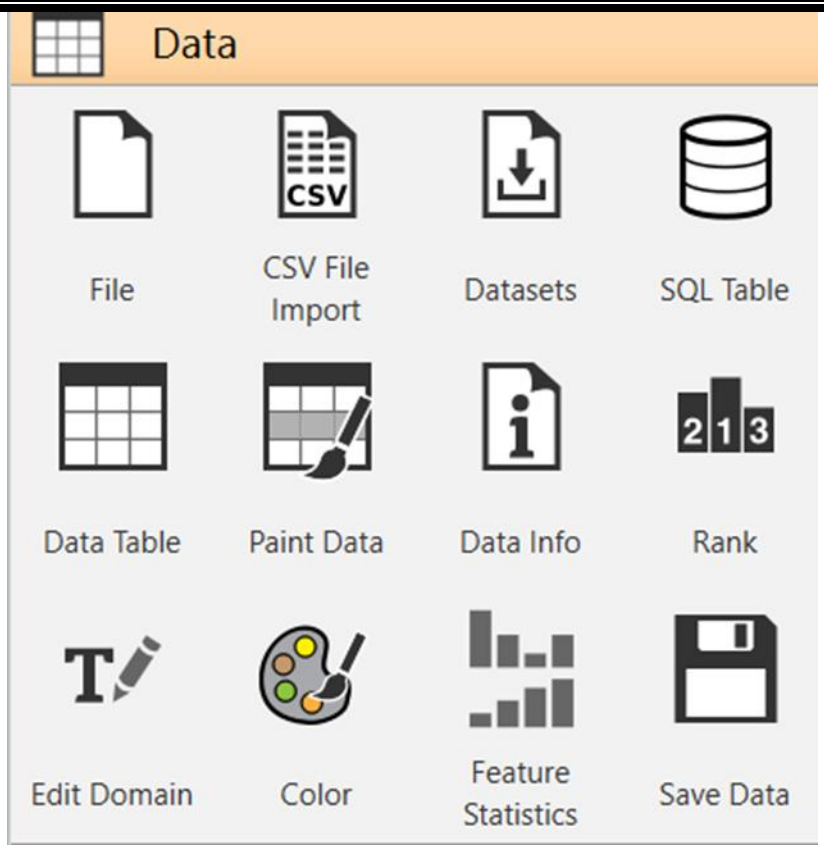
Toàn bộ quy trình được xây dựng và thực hiện trên phần mềm Orange Data Mining.

Bước 1: Nhập dữ liệu và xác định biến mục tiêu (Load Data and Define Target Variable)

Mục đích: Bước đầu tiên là đưa tập dữ liệu vào môi trường làm việc của Orange và chỉ định cột nào là biến mục tiêu mà mô hình cần dự đoán.

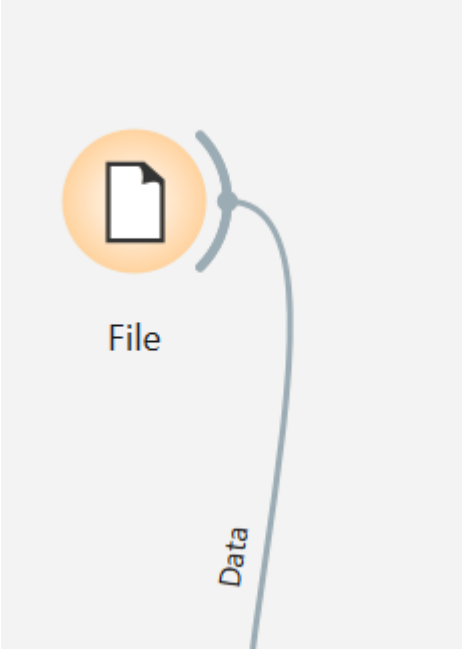
Thực hiện trên Orange:

Sử dụng **widget File** trên canvas của Orange.



Hình 2.1 Tool Data

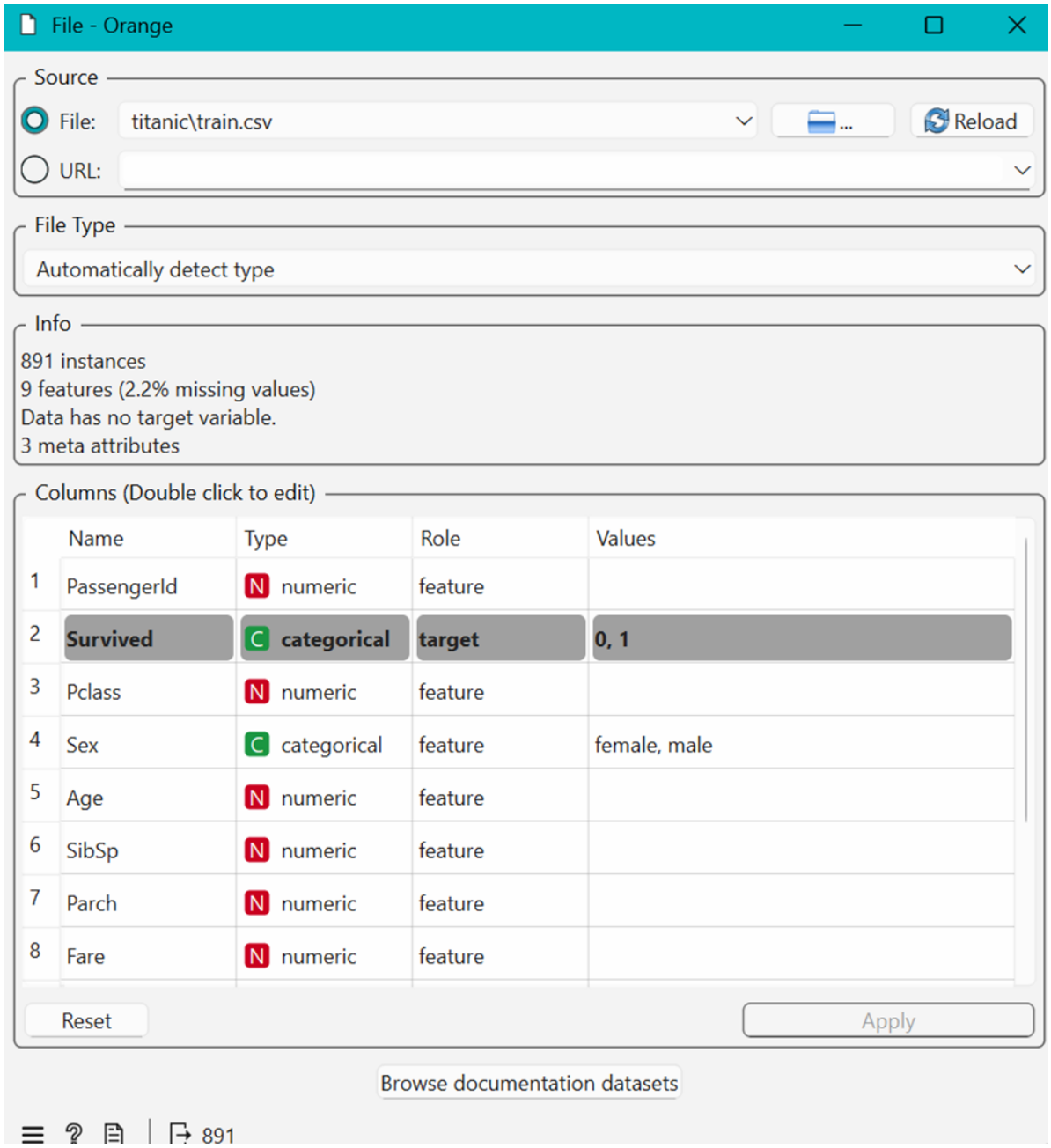
Bạn sẽ kéo và thả **widget File** từ thanh công cụ bên trái vào khu vực làm việc chính.



Hình 2.2 Kéo thả widget file

Nhấp đúp vào **widget File** để tải lên tập dữ liệu **Titanic Dataset**, thường là train.csv (dữ liệu được Kaggle cung cấp)

Trong cửa sổ cấu hình của **widget File**, bạn cần đảm bảo rằng cột **Survived** (biến nhị phân: 0 = không sống sót, 1 = sống sót) được thiết lập là **biến mục tiêu (Target)**. Các cột khác sẽ được mặc định là Features hoặc Meta.



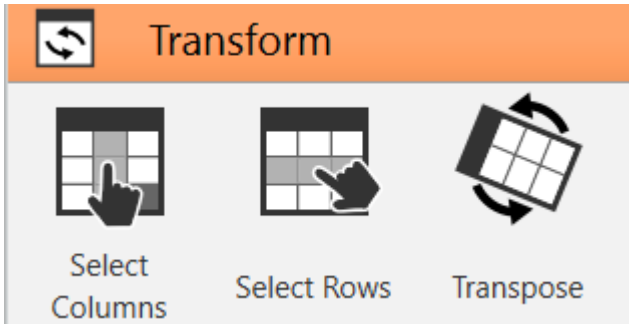
Hình 2.3 Cửa sổ widget

Bước 2: Lựa chọn thuộc tính (Select Columns):

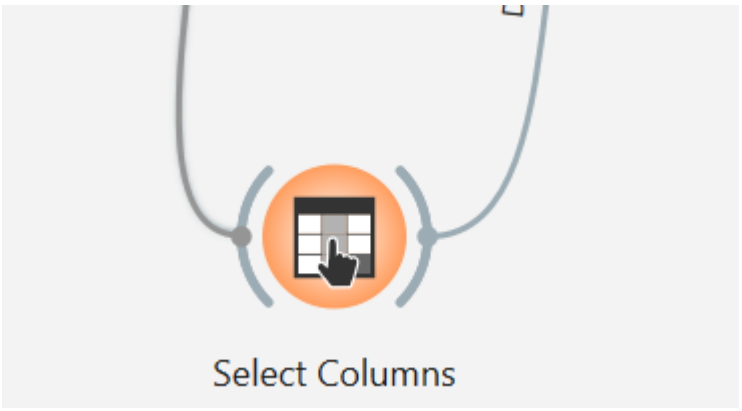
Mục đích: Để tối ưu hóa hiệu suất mô hình và giảm nhiễu, cần chọn lọc các thuộc tính đầu vào có ý nghĩa và loại bỏ các cột không cần thiết hoặc không mang nhiều thông tin.

Thực hiện trên Orange:

Kéo và thả **widget Select Columns** từ thanh công cụ vào canvas.

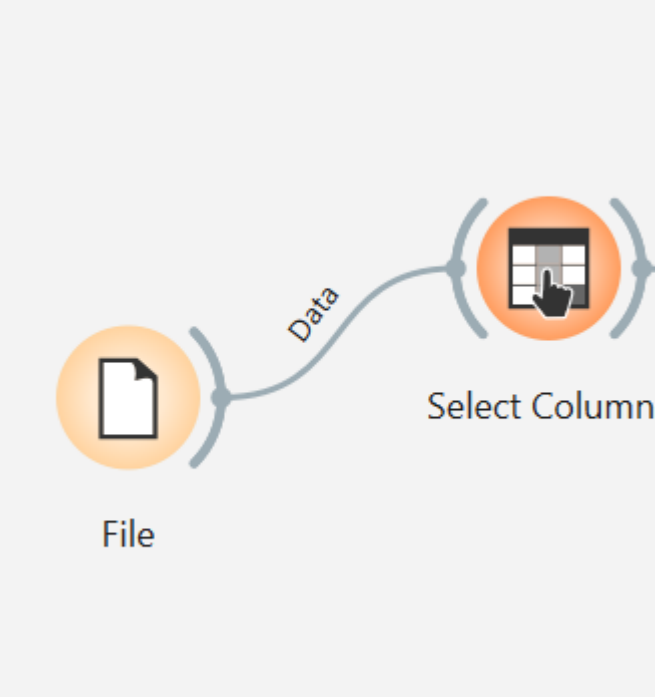


Hình 2.4 Transfrom



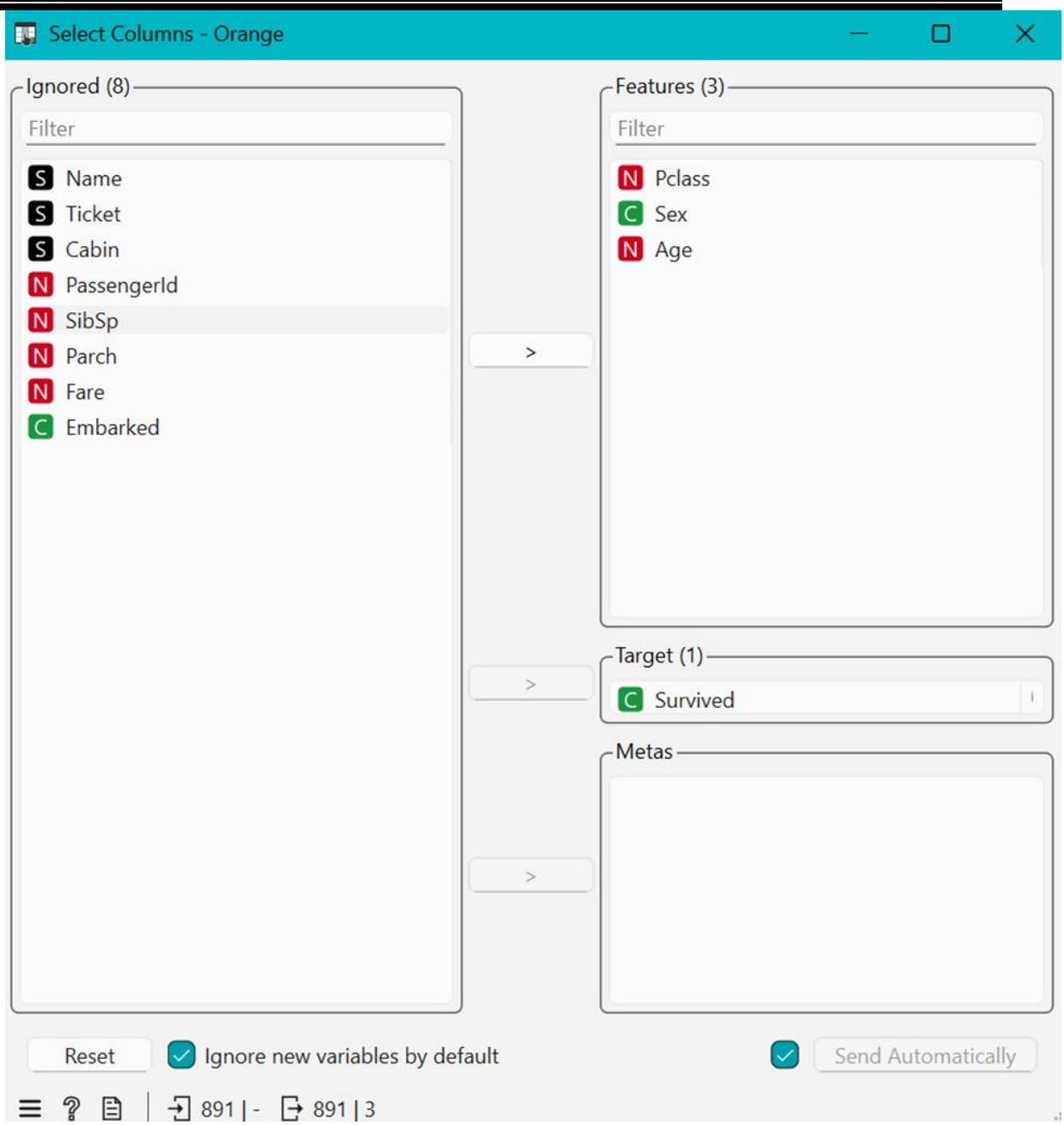
Hình 2.5 Select columns

Nối đầu ra dữ liệu từ **widget File** sang đầu vào của **widget Select Columns**.



Hình 2.6 Cấu hình

Nhấp đúp vào **widget Select Columns** để cấu hình.



Hình 2.7 Select column in orange

Chọn các thuộc tính đầu vào (Features): Nhóm đã chọn các thuộc tính sau vì chúng được cho là có ảnh hưởng rõ rệt đến khả năng sống sót của hành khách:

- **Pclass**: Hạng vé (1 = hạng nhất, 2 = hạng nhì, 3 = hạng ba) – phản ánh điều kiện kinh tế và có thể liên quan đến vị trí cabin trên tàu.
- **Sex**: Giới tính (nam/nữ) – một yếu tố quan trọng trong các quy tắc cứu hộ ("phụ nữ và trẻ em trước").

- **Age**: Tuổi hành khách – cũng là một yếu tố ảnh hưởng đến khả năng sống sót.

Giữ cột mục tiêu (Target): Đảm bảo cột **Survived** vẫn được giữ làm mục tiêu.

Loại bỏ các cột không cần thiết: Các cột sau đây bị loại bỏ vì chúng không có giá trị phân loại rõ ràng, chứa quá nhiều giá trị thiếu hoặc chỉ là mã định danh:

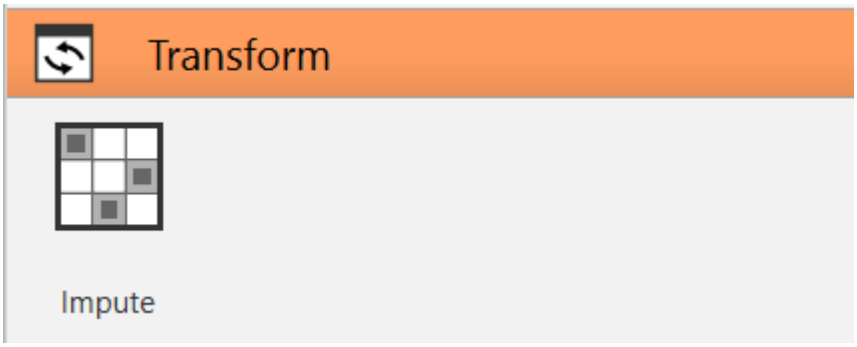
- **Name**: Tên hành khách (thường bị loại bỏ do không mang giá trị phân lớp).
- **Ticket**: Mã số vé (không có giá trị phân loại rõ ràng, thường bị bỏ).
- **Cabin**: Mã phòng (rất nhiều giá trị thiếu – thường bị loại).
- **PassengerId**: ID định danh từng hành khách (không dùng trong huấn luyện).
- **SibSp**: Số anh chị em/vợ/chồng đi cùng.
- **Parch**: Số cha mẹ/con cái đi cùng.
- **Fare**: Giá vé hành khách đã trả.
- **Embarked**: Cảng nơi hành khách lên tàu.

Bước 3: Xử lý giá trị thiếu (Impute Missing Values)

Mục đích: Dữ liệu thực tế thường chứa các giá trị bị khuyết, nếu không xử lý sẽ ảnh hưởng đến quá trình huấn luyện mô hình. Bước này giúp điền các giá trị hợp lý vào chỗ trống.

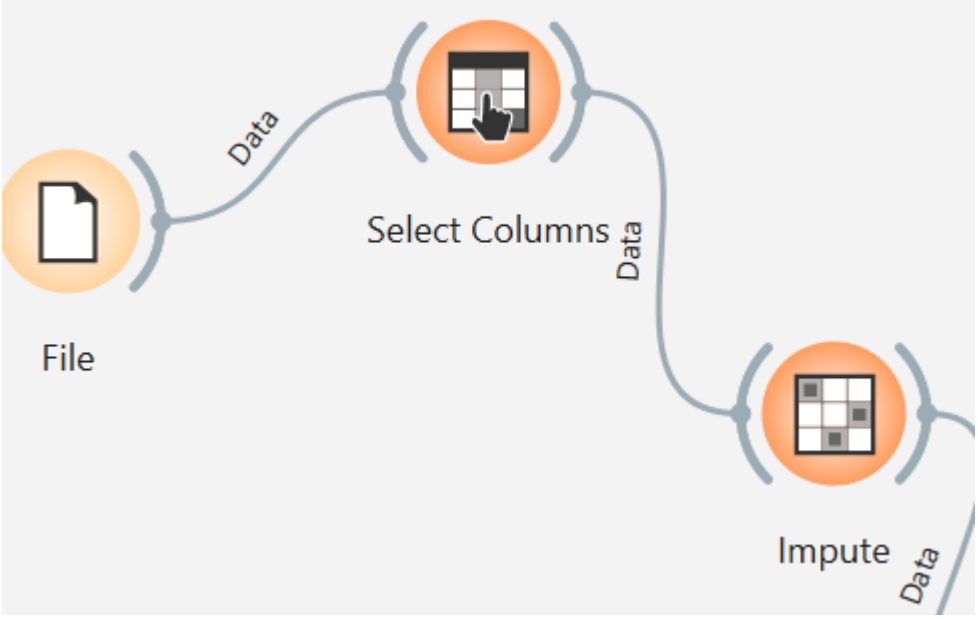
Thực hiện trên Orange:

Kéo và thả **widget Impute** từ thanh công cụ vào canvas



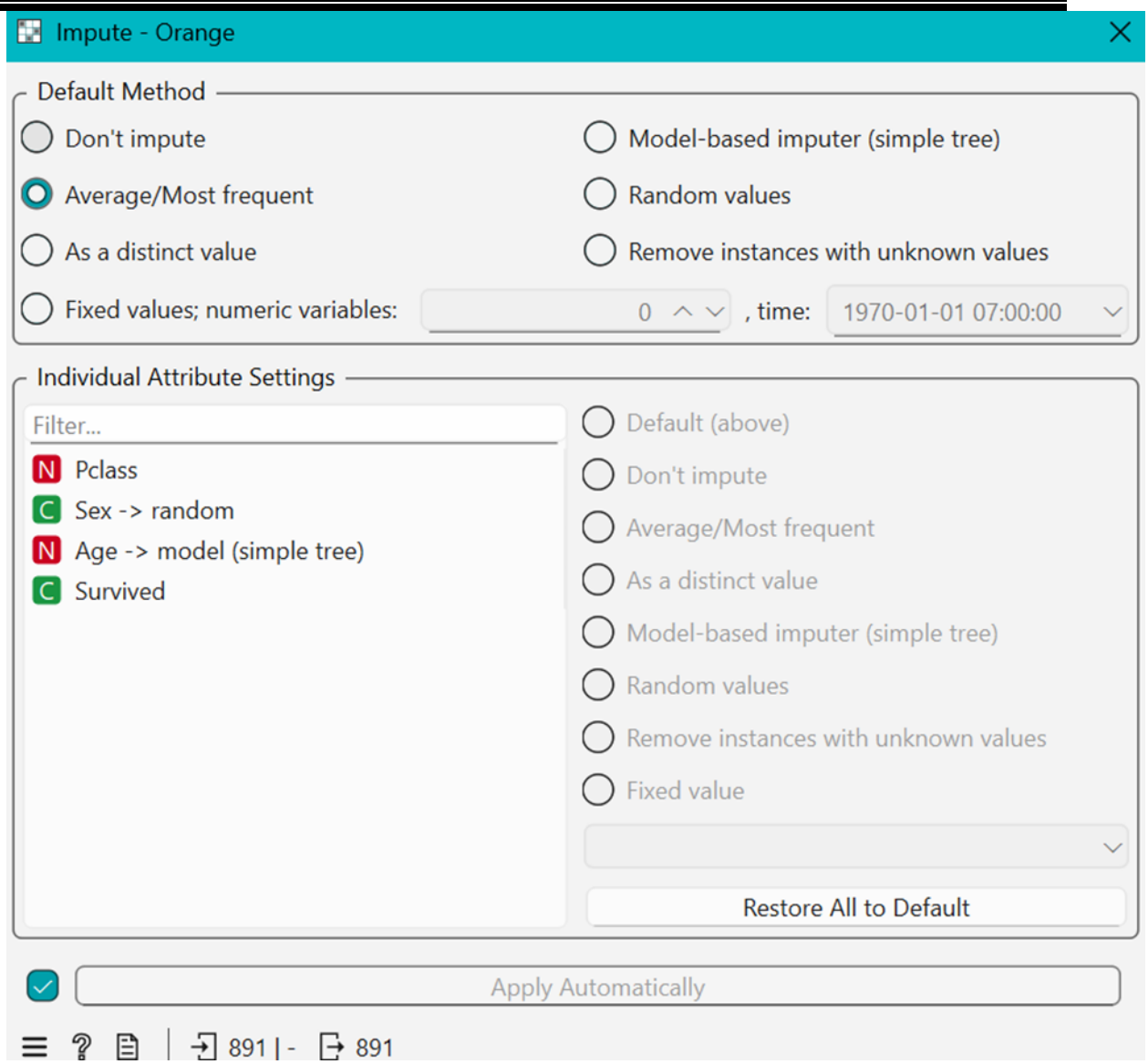
Hình 2.8 Inpute

Nối đầu ra dữ liệu từ **widget Select Columns** sang đầu vào của **widget Impute**.



Hình 2.9 Nối các widget

Nhấp đúp vào **widget Impute** để cấu hình.



Hình 2.10 Giao diện widget

Xử lý cột Age: Cột Age có thể chứa giá trị bị thiếu. Nhóm đã chọn phương pháp **điền giá trị trung bình (mean)** của cột để thay thế các giá trị này. Điều này giúp duy trì phân phối tổng thể của dữ liệu.

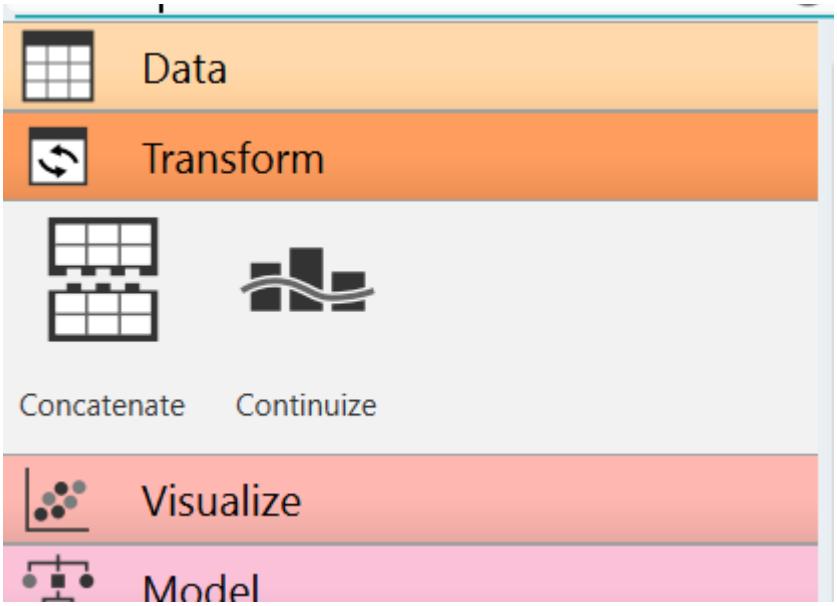
Xử lý cột Sex: Cột sex có thể chứa giá trị bị thiếu. Nhóm đã chọn phương pháp **chọn giá trị ngẫu nhiên (Radom)** của cột để thay thế các giá trị bị thiếu. Điều này giúp các giới tính được chọn lọc ngẫu nhiên cho tổng thể dữ liệu

Bước 4: Chuyển đổi dữ liệu (Continuize / Mã hóa dữ liệu phân loại)

Mục đích: Một số thuật toán học máy (như kNN) yêu cầu tất cả dữ liệu đầu vào phải ở dạng số để tính toán khoảng cách hoặc thực hiện các phép toán khác. Bước này giúp chuyển đổi các biến phân loại (categorical) thành định dạng số phù hợp.

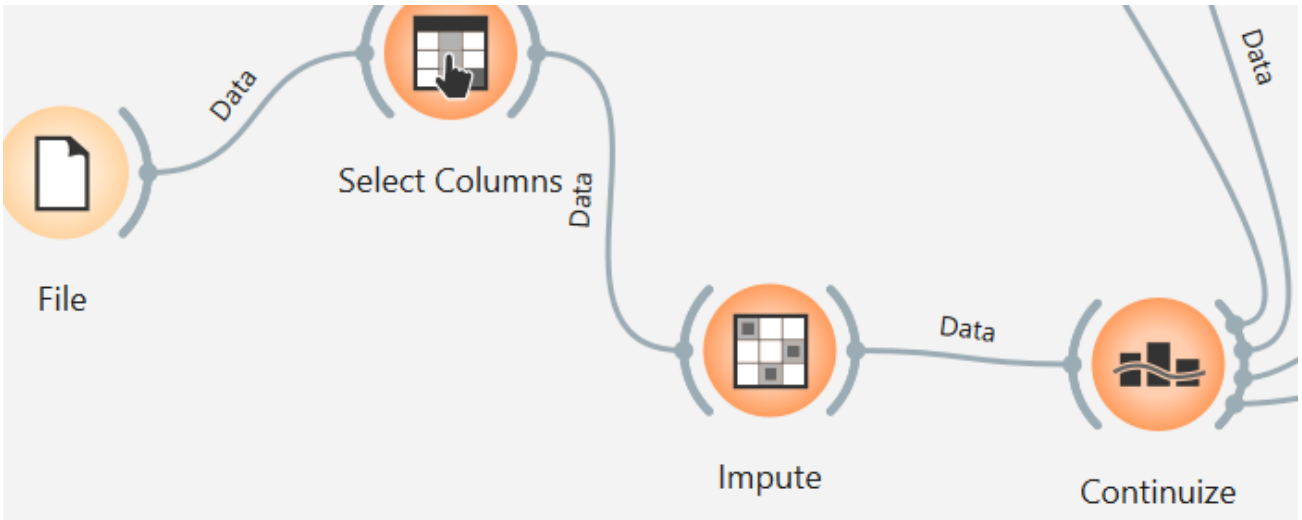
Thực hiện trên Orange:

Kéo và thả **widget Continuize** từ thanh công cụ vào canvas.



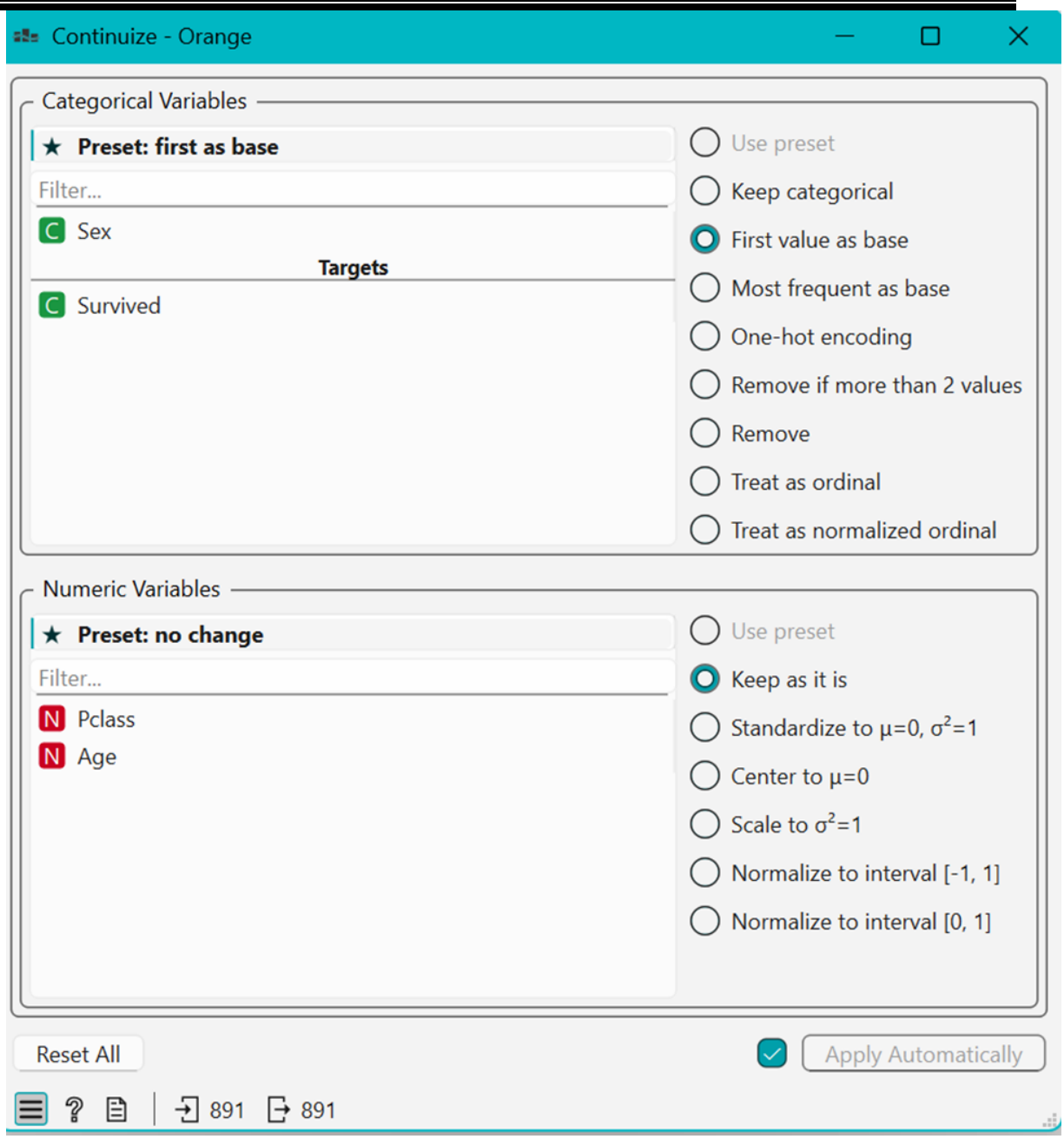
Hình 2.11 Widget

Nối đầu ra dữ liệu từ **widget Impute** sang đầu vào của **widget Continuize**.



Hình 2.12 Nối widget

Nhấp đúp vào **widget Continuize** để cấu hình.



Hình 2.13 Giao diện widget

Cột Sex: Cột Sex (giới tính: Nam/Nữ) là một biến phân loại. Để chuyển đổi sang dạng số, Orange sẽ mã hóa biến này. Ví dụ, thiết lập **"First value as base"** sẽ mã hóa male thành 0 và female thành 1. Đây là mã hóa nhị phân giúp mô hình hiểu sự khác biệt giữa hai giới tính.

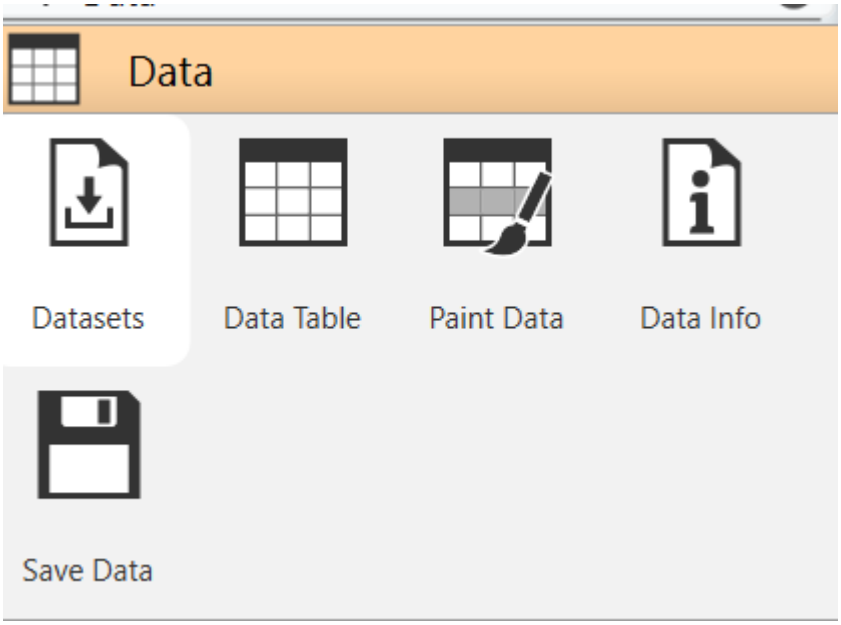
Cột Pclass và Age: Các cột này đã ở dạng số hoặc được xử lý ở bước trước. Đối với Pclass (Hạng vé), mặc dù là số nguyên, nó có thể được coi là biến thứ tự hoặc phân loại. Trong trường hợp này, bạn có thể chọn **"Keep as it is"** nếu muốn giữ nguyên định dạng số của chúng hoặc cấu hình thêm nếu muốn xử lý chúng như biến phân loại nhị phân (ví dụ: tạo các biến giả - one-hot encoding).

Bước 5: Khảo sát thông tin và trực quan hóa dữ liệu (Data Information and Visualization)

Mục đích: Sau các bước tiền xử lý, việc khảo sát lại dữ liệu là cần thiết để đảm bảo rằng các thay đổi đã được áp dụng đúng cách và dữ liệu đã sẵn sàng cho quá trình xây dựng mô hình.

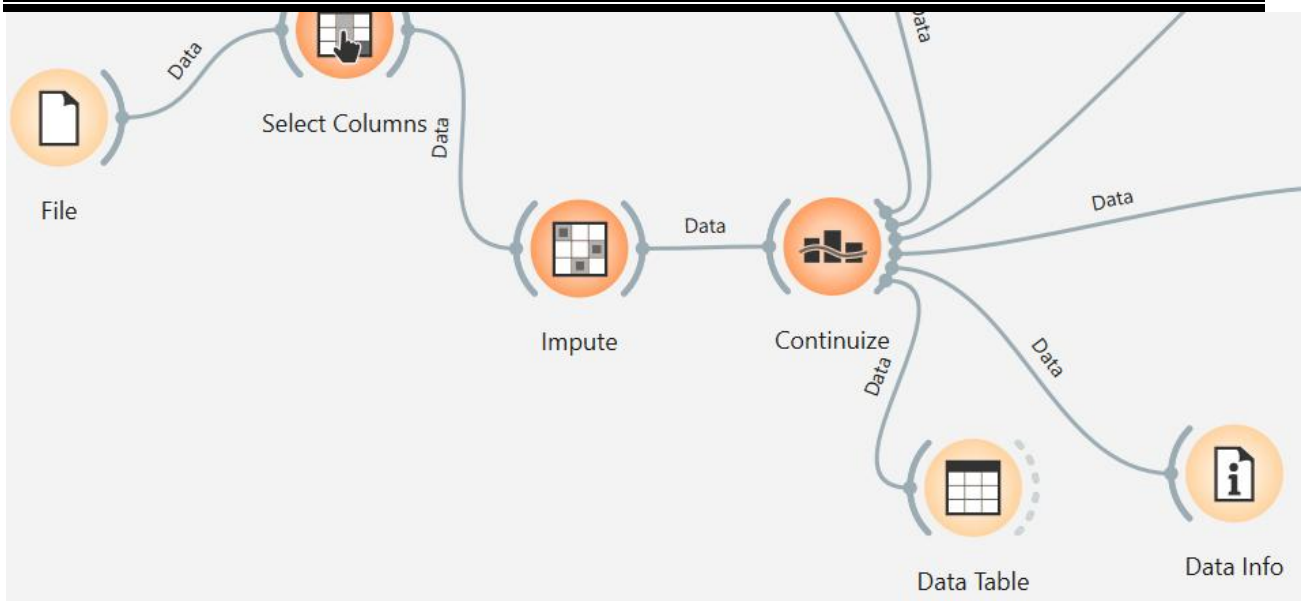
Thực hiện trên Orange:

Kéo và thả **widget Data Info** và **Data Table** vào canvas



Hình 2.14 widget

Nối đầu ra dữ liệu từ **widget Continuize** (hoặc bước tiền xử lý cuối cùng) sang đầu vào của **widget Data Info** và **Data Table**.



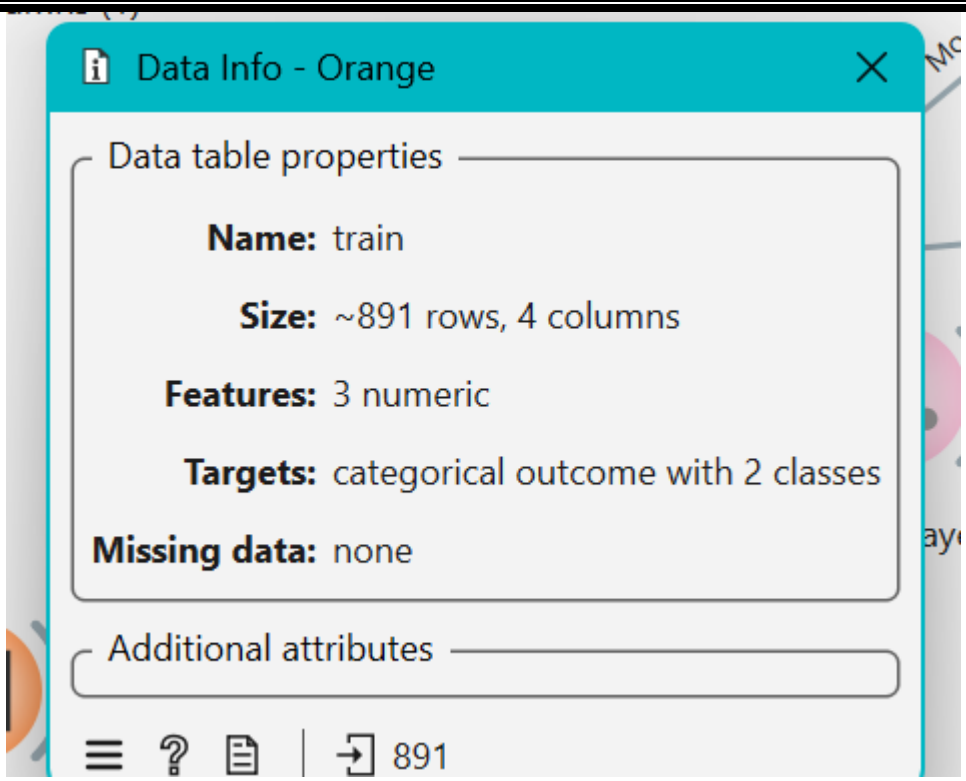
Hình 2.15 Nối widget

Với Data Info: Nhấp đúp vào **widget Data Info** để xem thống kê tổng quan về dữ liệu như:

Số lượng cột, số lượng dòng.

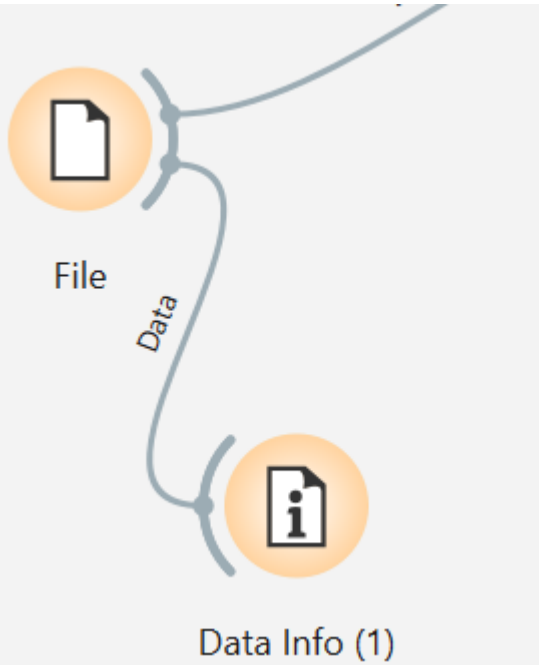
Kiểu dữ liệu của từng cột (rời rạc, liên tục).

Tỷ lệ giá trị bị thiếu (lúc này, tỷ lệ giá trị thiếu của Age và Embarked nếu có sẽ được giảm hoặc bằng 0).

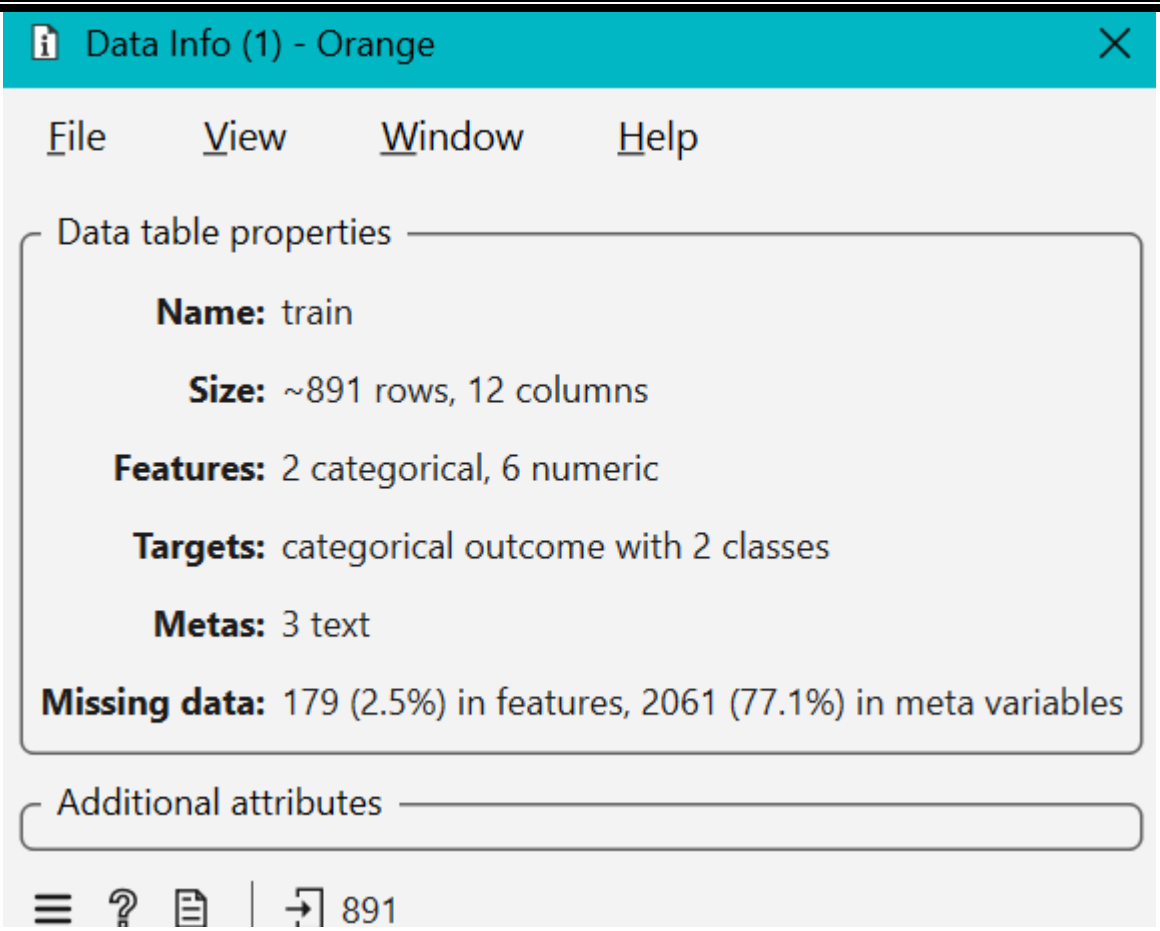


Hình 2.16 Data info

So với **Data Info được nối với File** chứa dữ liệu trước khi tiền xử lý thì độ Missing Value đã cải thiện rất đáng kể.

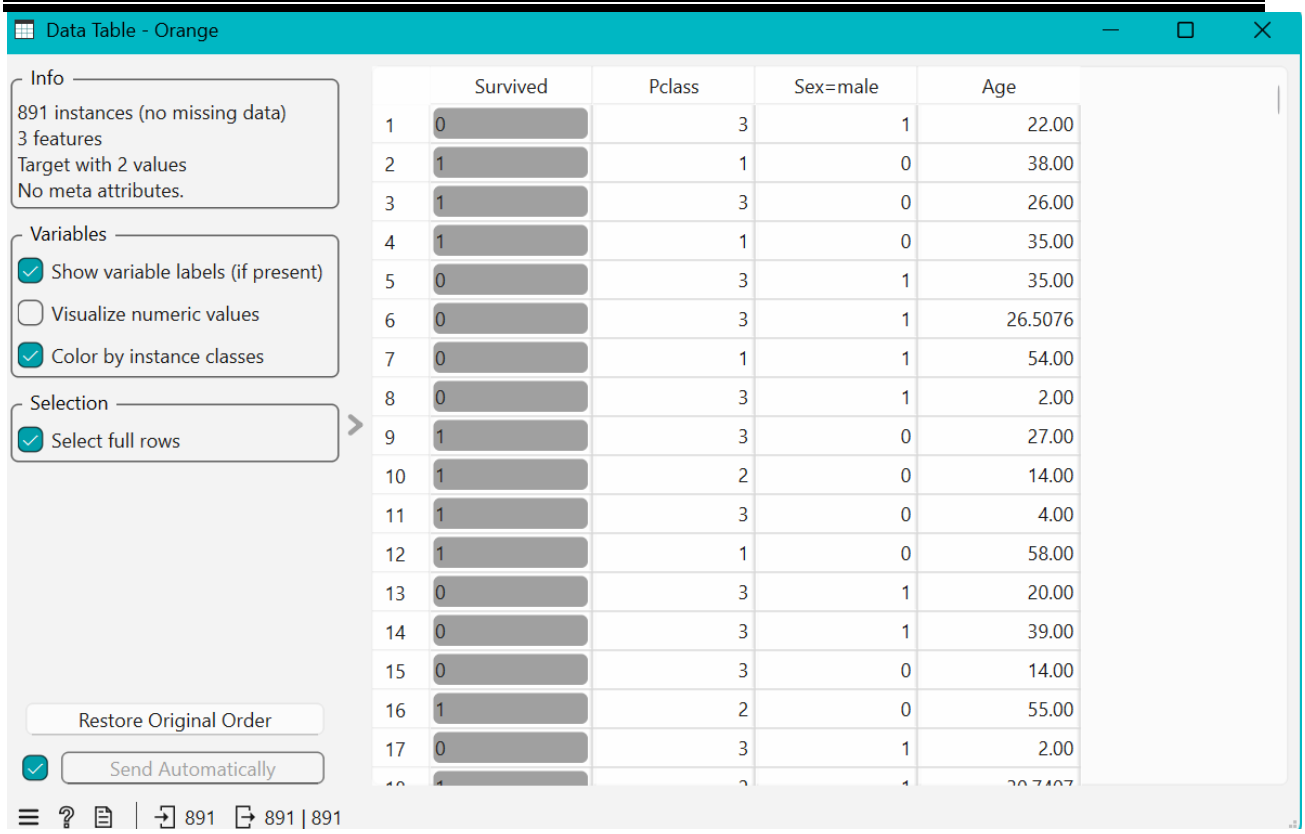


Hình 2.17 Nối data info



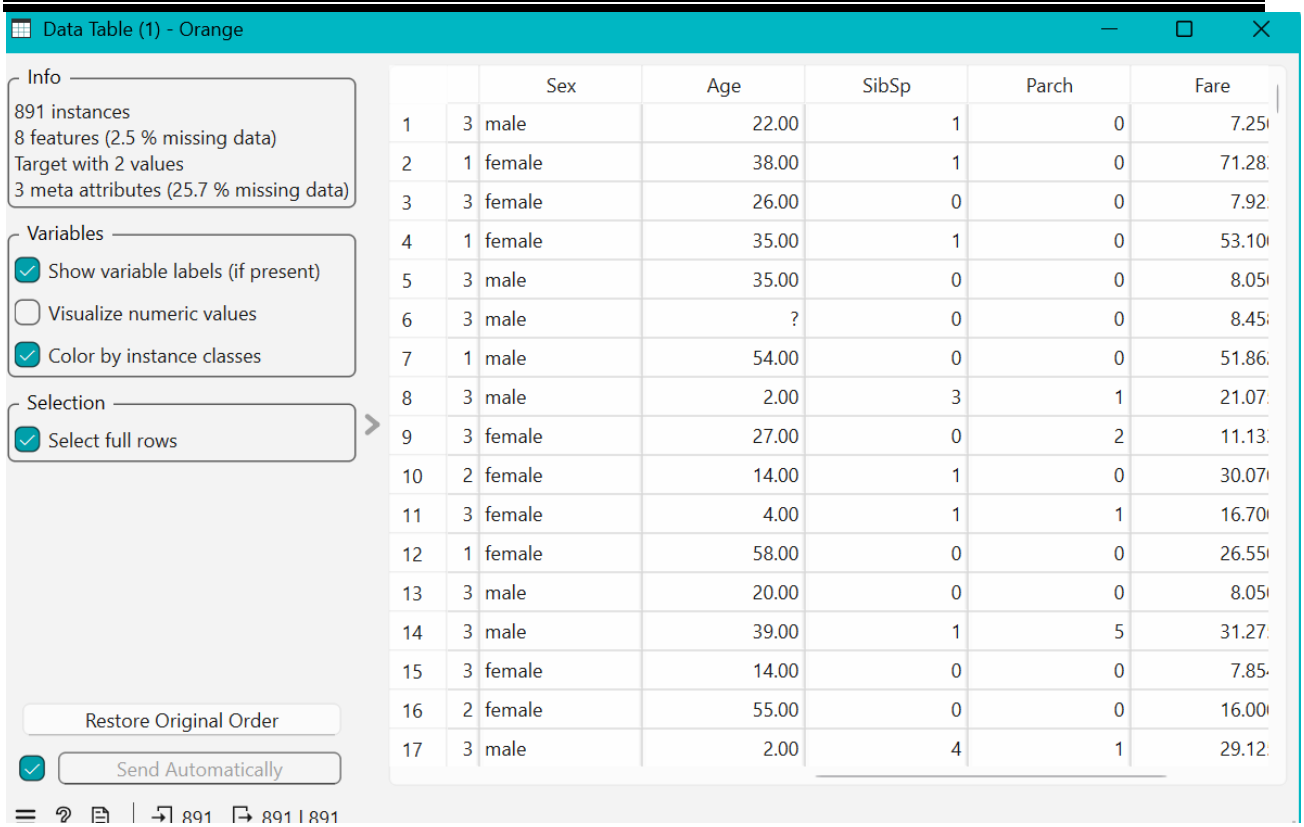
Hình 2.18 Data info

Với Data Table: Nhấp đúp vào **widget Data Table** để quan sát trực tiếp bảng dữ liệu đã được tiền xử lý.



Hình 2.19 Data table

Như ta đã thấy dữ liệu cột Sex đã được Mã hóa về 0 và 1 so với Dữ liệu ban đầu được nối trực tiếp với File thay vì phải tiền xử lý dữ liệu.



		Sex	Age	SibSp	Parch	Fare
1	3	male	22.00	1	0	7.25
2	1	female	38.00	1	0	71.28
3	3	female	26.00	0	0	7.92
4	1	female	35.00	1	0	53.10
5	3	male	35.00	0	0	8.05
6	3	male	?	0	0	8.45
7	1	male	54.00	0	0	51.86
8	3	male	2.00	3	1	21.07
9	3	female	27.00	0	2	11.13
10	2	female	14.00	1	0	30.07
11	3	female	4.00	1	1	16.70
12	1	female	58.00	0	0	26.55
13	3	male	20.00	0	0	8.05
14	3	male	39.00	1	5	31.27
15	3	female	14.00	0	0	7.85
16	2	female	55.00	0	0	16.00
17	3	male	2.00	4	1	29.12

Hình 2.20 Data table

CHƯƠNG 3. THỰC NGHIỆM MÔ HÌNH TRÊN ORANGE

Ở chương 3 này chúng em xin thực hiện đầu tiên ở mỗi Mô Hình là các tính toán bằng công thức hay còn gọi là Thực nghiệm thủ công sau đó mới tiếp tục với Mô hình trên Orange với dữ liệu sau khi đã qua tiền xử lý được trình bày ở Chương 2 ở trên.

3.1. Mô hình kNN (k Nearest Neighbor)

Mô hình kNN hoạt động bằng cách tìm k điểm gần nhất trong tập huấn luyện để dự đoán nhãn cho điểm mới. Ý tưởng chính là dữ liệu tương tự thường nằm gần nhau trong không gian.

Thực nghiệm thủ công

Dữ liệu sử dụng:

Tập huấn luyện (tập train.csv) lấy 5 mẫu đầu tiên:

Table 2 Dữ liệu mẫu

STT	Pclass	Sex (Nam=0/Nữ=1)	Age	Survived
0	3	0	22.0	0
1	1	1	38.0	1
2	3	1	26.0	1
3	1	1	35.0	1
4	3	0	35.0	0

Điểm cần dự đoán (từ tập test.csv)

Table 3 Điểm cần dự đoán

Pclass	Sex	Age
3	0	34.5

Tiền xử lý dữ liệu:

Trong bước này, ta chuẩn bị dữ liệu sao cho phù hợp với việc tính khoảng cách trong mô hình **k-NN**. Các bước cụ thể như sau:

Chọn các thuộc tính đặc trưng:

Từ bộ dữ liệu Titanic gốc, có rất nhiều thuộc tính như: Name, Ticket, Cabin, Fare, Embarked, v.v.

Tuy nhiên, để đơn giản hóa việc tính toán thủ công, ta chỉ chọn 3 thuộc tính đầu vào có ảnh hưởng rõ đến khả năng sống sót:

Table 4 Mô tả

Thuộc tính	Mô tả	Kiểu dữ liệu	Ghi chú
Pclass	Hạng vé (1 = cao, 3 = thấp)	Số nguyên	Giá trị nhỏ hơn thường sống cao hơn
Sex	Giới tính	Chuỗi	Cần chuyển đổi sang số
Age	Tuổi hành khách	Số thực	Dữ liệu liên tục, có thể thiếu

Mã hóa cột Sex

Mô hình kNN yêu cầu dữ liệu số để tính toán. Do đó, cột Sex dạng chữ phải được chuyển thành số:

Table 5 Mã hóa

Sex	Mã hóa
male	0
female	1

Đây là mã hóa **nhị phân** giúp mô hình hiểu sự khác biệt giữa nam và nữ.

Xử lý thiếu dữ liệu

Trong cột Age, có thể một số dòng bị thiếu. Trong thực nghiệm này, ta chỉ chọn các dòng **không có giá trị thiếu (dropna)** để tính toán cho đơn giản.

Nếu áp dụng toàn bộ dữ liệu, ta nên **điền trung bình/median** cho cột Age. Không chuẩn hóa đặc trưng

Trong trường hợp thực nghiệm này, ta **không chuẩn hóa** (normalization) các giá trị đầu vào, vì:

- Số lượng mẫu ít

- Khoảng giá trị giữa các đặc trưng không quá chênh lệch
- Dễ quan sát và mô phỏng

Công thực tính khoảng cách Euclidean

Cho điểm A = (a1, a2, a3); B = (b1, b2, b3):

Distance(A, B) = $\sqrt{(a1 - b1)^2 + (a2 - b2)^2 + (a3 - b3)^2}$

Tính khoảng cách từ điểm test đến từng điểm train

Gía trị điểm test:

-> Pclass = 3, Sex = 0, Age = 34.5

Table 6 Tính KNN

STT	Pclass	Sex	Age	Survived	Distance (khoảng cách)	Công thức chi tiết
0	3	0	35.0	0	0.500	$\sqrt{(3-3)^2 + (0-0)^2 + (34.5-35)^2}$ $=\sqrt{(0)^2 + (0)^2 + (0.25)^2}$ = **0.5**
1	1	1	35.0	1	2.291	$\sqrt{(3-1)^2 + (0-1)^2 + (34.5-35)^2}$ $= \sqrt{(4) + (1) + (0.25)}$ = √5.25 = **2.29**
2	1	1	38.0	1	4.153	$\sqrt{(3-1)^2 + (0-1)^2 + (34.5-38)^2}$ $=\sqrt{(4) + (1) + (12.25)}$ = √17.25 = **4.15**
3	3	1	26.0	1	8.559	$\sqrt{(3-3)^2 + (0-1)^2 + (34.5-26)^2}$ $=\sqrt{(0) + (1) + (75.25)}$

						= √73.25 = **8.56**
4	3	0	22.0	0	12.500	$\sqrt{(3-3)^2+(0-0)^2+(34.5-22)^2}$ $=\sqrt{(0)+(0)+(156.25)}$ =√156.25 = **12.5**

Chọn ra k = 3 điểm gần nhất

Table 7 Dữ đoán

STT	Pclass	Sex	Age	Survived	Distance
0	3	0	35.0	0	0.500
1	1	1	35.0	1	2.291
2	1	1	38.0	1	4.153

Số hành khách **sống sót** (Survived = 1): **2**

Số hành khách **không sống sót** (Survived = 0): **1**

Mô hình sử dụng phương pháp bỏ phiếu đa số (majority vote):

Prediction=mode([0,1,1]) = 1

Dự đoán cuối cùng cho điểm test là:

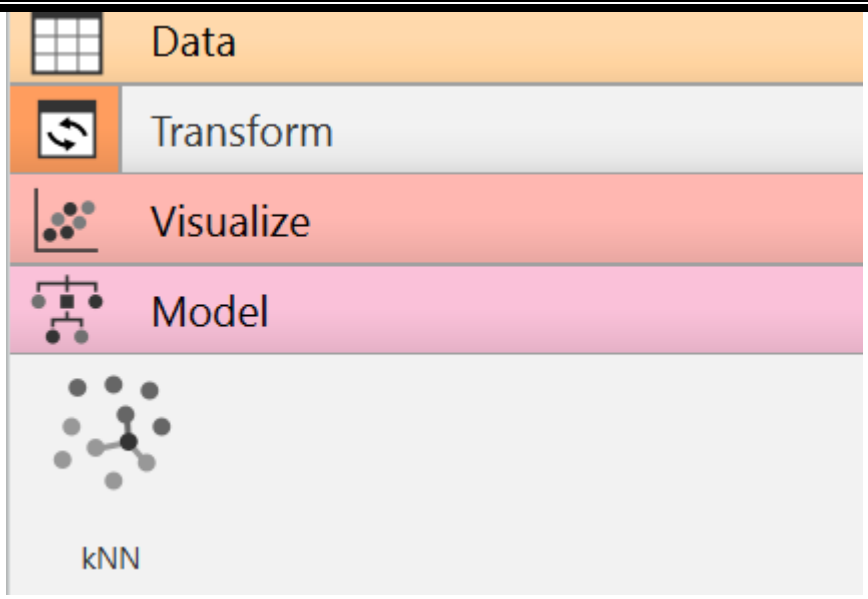
1 – Hành khách sẽ sống sót

Thực nghiệm trên Orange:

Sau khi dữ liệu đã được tiền xử lý ở Chương 2 (Load Data, Select Columns, Impute, Continuize), ta sẽ thiết lập mô hình kNN và đánh giá.

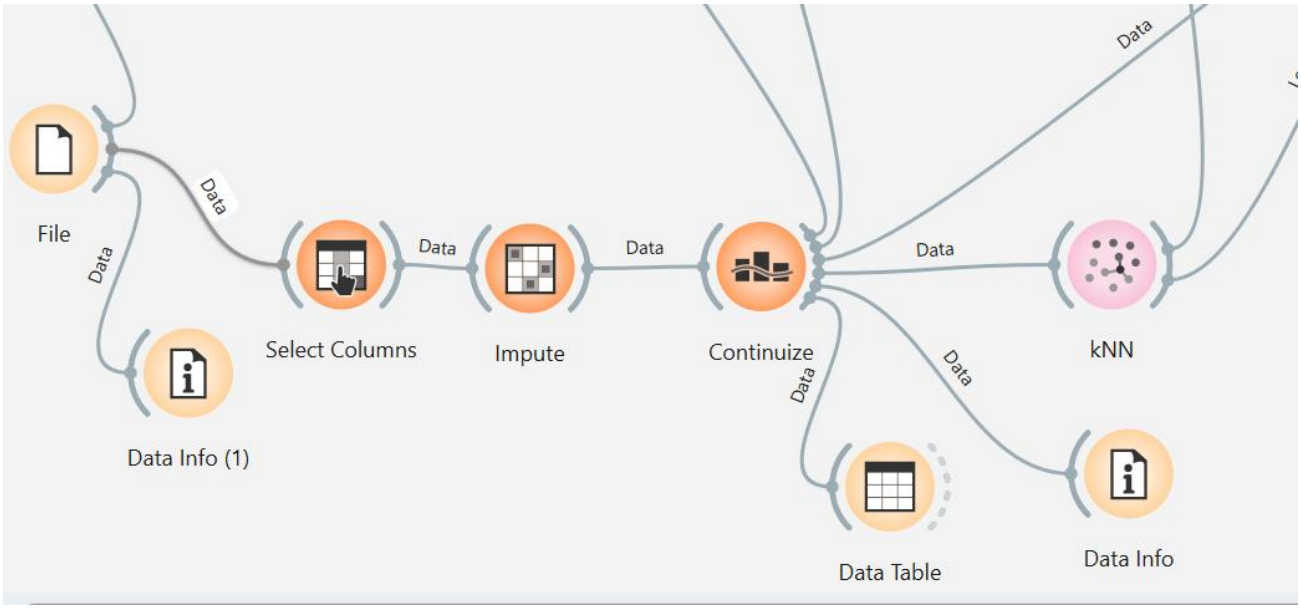
Bước 1: Thiết lập mô hình kNN:

Kéo và thả **widget kNN** từ nhóm "Model" vào canvas.



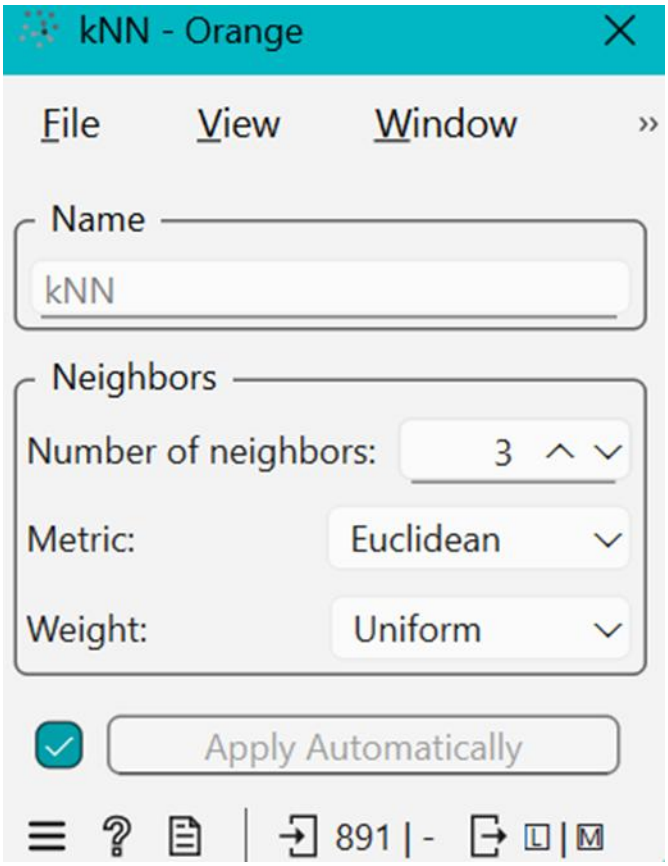
Hình 3.1 Widget

Nối đầu ra dữ liệu đã tiền xử lý từ **widget Continuize** (hoặc Impute nếu không dùng Continuize cho các biến phân loại) đến đầu vào của **widget kNN**.



Hình 3.2 Nối widget

Nhấp đúp vào widget kNN để cấu hình các tham số. Mặc dù tài liệu không nêu rõ các tham số cụ thể của kNN được chọn trong Orange, kNN thường có các tham số như số lượng k láng giềng, phương pháp đo khoảng cách (metric) và phương pháp gán trọng số (weight).



Hình 3.3 widget KNN

- **metric**: Bao gồm các phương pháp tính khoảng cách như Euclidean, Manhattan. Euclidean là phổ biến nhất.
- **weight**: Bao gồm các phương pháp gán trọng số cho láng giềng, ví dụ như uniform (tất cả các láng giềng có trọng số như nhau) hoặc distance (láng giềng gần hơn có trọng số cao hơn).

Như trên hình trên thì nhóm chúng em chọn số lượng láng giềng K = 3 với Metrics là phương pháp **Euclidean** giống với thực nghiệm thủ công và **Weight** với phương pháp gán trọng số Uniform.

3.2. Mô hình Cây quyết định (Decision tree)

Mô hình Cây quyết định phân loại dữ liệu bằng cách xây dựng một cấu trúc cây phân cấp dựa trên các quy tắc phân chia. Mỗi nút trong cây đại diện cho một thuộc tính, mỗi nhánh đại diện cho một giá trị (hoặc khoảng giá trị) của thuộc tính đó, và các nút lá đại diện cho lớp dự đoán. Quá trình xây dựng cây liên quan đến việc tìm kiếm các điểm phân chia tốt nhất dựa trên độ "thuần khiết" của dữ liệu.

Thực nghiệm thủ công:

Dữ liệu sử dụng: Để minh họa, ta vẫn sử dụng 5 mẫu đầu tiên từ tập huấn luyện (train.csv) và một điểm cần dự đoán từ tập kiểm tra (test.csv) như đã dùng cho kNN.

Tập huấn luyện (tập train.csv lấy 5 mẫu đầu tiên):

Table 8 Dữ liệu mẫu

STT	Pclass	Sex (Nam=0/Nữ=1)	Age	Survived
0	3	0	22.0	0
1	1	1	38.0	1
2	3	1	26.0	1
3	1	1	35.0	1
4	3	0	35.0	0

Điểm cần dự đoán (từ tập test.csv):

Table 9 Điểm cần dự đoán

Pclass	Sex	Age
3	0	34.5

Tiền xử lý dữ liệu: Các bước tiền xử lý tương tự như kNN để chuẩn bị dữ liệu đầu vào:

Chọn các thuộc tính đặc trưng: Để đơn giản hóa việc tính toán thủ công, ta chọn 3 thuộc tính đầu vào có ảnh hưởng rõ đến khả năng sống sót: Pclass, Sex, Age. Cột mục tiêu là Survived.

Mã hóacột Sex:Cột Sex dạng chữ được chuyển thành số: male thành 0,female thành 1

Xử lý thiếu dữ liệu: Trong thực nghiệm này, ta chọn các dòng không có giá trị thiếu (dropna) để tính toán cho đơn giản.

Không chuẩn hóa đặc trưng: Ta không chuẩn hóa (normalization) các giá trị đầu vào để dễ quan sát và mô phỏng, do số lượng mẫu ít và khoảng giá trị giữa các đặc trưng không quá chênh lệch

Các khái niệm cơ bản cho Cây quyết định:

Để xây dựng cây quyết định thủ công, chúng ta cần hiểu các khái niệm về độ không tinh khiết (Impurity) và độ lợi thông tin (Information Gain) (hoặc Gini Gain):

Gini Impurity (Độ không tinh khiết Gini): Đo lường khả năng một phần tử được chọn ngẫu nhiên từ tập dữ liệu bị phân loại sai nếu nó được gán nhãn ngẫu nhiên theo phân phối nhãn trong tập dữ liệu đó. Giá trị càng thấp, tập dữ liệu càng "thuần khiết".

Công thức Gini Impurity cho một nút t với C lớp:

$$G(t) = 1 - \sum_{i=1}^{C} \left(P(i \mid t|)\right)^2$$

Ý nghĩa:

*CC*C: Số lượng lớp (categories) trong bài toán phân loại.

$G\left(P(i \mid t|)\right)^2$: Tỷ lệ phần trăm (xác suất) các mẫu thuộc lớp *i* tại nút *t*

Giải thích bằng ví dụ:

Giả sử một nút có 3 lớp:

50% thuộc lớp A

30% thuộc lớp B

20% thuộc lớp C

Khi đó:

$$G(t) = 1 - (0.5)^2 - (0.3)^2 - (0.2)^2 = 1 - 0.25 - 0.09 - 0.04 = 0.62$$

Gini = 0: nút hoàn toàn thuần khiết (chỉ có 1 lớp).

Gini càng cao → nút càng hỗn loạn → cần phân chia tiếp.

Information Gain (Độ lợi thông tin)**:** Đo lường mức độ giảm độ không tinh khiết sau khi chia dữ liệu theo một thuộc tính nhất định. Chúng ta chọn thuộc tính nào mang lại Information Gain cao nhất để chia nút.

Công thức Information Gain cho thuộc tính A khi chia tập D:

$$IG(A) = G(D) - \sum_{v \epsilon Values(A)} |D_v||D| \,.\, G(D_v)$$

Table 10 Mô tả ký hiệu

Ký hiệu	Ý nghĩa
IG(A)	Độ lợi thông tin của thuộc tính A
G(D)	Gini Impurity của toàn bộ tập D (trước khi chia)
Values(A)	Tập các giá trị có thể có của thuộc tính A (vd: Nam/Nữ, Cao/Thấp,...)
Dv	Tập con các mẫu có thuộc tính A = v
(	D_v
(	D
G(Dv)	Gini Impurity của tập con *Dv*

Các bước xây dựng cây quyết định thủ công (Áp dụng cho 5 mẫu dữ liệu):

Bước 1. Tính Gini Impurity ban đầu của tập Survived:

Tổng số mẫu: 5

Số mẫu Survived = 0: 2

Số mẫu Survived = 1: 3

Tính xác suất:$P(Survived = 0) = \frac{2}{5} = 0.4$, $P(Survived = 1) = \frac{3}{5} = 0.6$

Tính Gini Impurity gốc:$G_{root} = 1 - (0.4)^2 - (0.6)^2 = 1 - 0.16 - 0.36 = 0.48$

Bước 2. Tìm thuộc tính tốt nhất để chia:

Thuộc tính Sex

Nếu Sex = 0 (Nam): 2 mẫu, cả 2 đều Survived = 0

$$G(Sex = 0) = 1 - (1)^2 - (0)^2 = 0$$

Nếu Sex = 1 (Nữ): 3 mẫu, cả 3 đều Survived = 1

$$G(Sex = 1) = 1 - (0)^2 - (1)^2 = 0$$

Tính Information Gain:

$$IG(Sex) = G(root) - \left(\frac{2}{5}.G(Sex = 0) + \frac{3}{5}.G(Sex = 1)\right)$$

$$= IG(Sex) = 48 - \left(\frac{2}{5}.0 + \frac{3}{5}.0\right) = 0.48 - 0 = 0.48$$

Thuộc tính Pclass:

Pclass = 1: 2 mẫu, Survived = 1 cả hai

$$G(Pclass = 1) = 1 - 0^2 - 1^2 = 0$$

Pclass = 3: 3 mẫu, 2 mẫu Survived = 0, 1 mẫu Survived = 1

$$G(Pclass = 3) = 1 - \left(\frac{2}{3}\right)^2 - \left(\frac{1}{3}\right)^2 = 1 - \frac{4}{9} - \frac{1}{9} = \frac{4}{9}$$

Tính Information Gain:

$$IG(Pclass) = 0.48 - \left(\frac{2}{5}.0 + \frac{3}{5}.\left(\frac{4}{9}\right)\right) = 0.48 - (0 + 0.2664) = 0.2136$$

Thuộc tính Age (liên tục – cần chọn ngưỡng):

Sắp xếp theo tuổi và nhãn Survived:

Table 11 Mô tả sắp xếp

Age	Survived
22.0	0
26.0	1
35.0	1
35.0	0
38.0	1

Ngưỡng chia khả thi:

(22+26)/2=24.0(22 + 26)/2 = 24.0(22+26)/2=24.0

(26+35)/2=30.5(26 + 35)/2 = 30.5(26+35)/2=30.5

(35+35)/2=35.0(35 + 35)/2 = 35.0(35+35)/2=35.0

(35+38)/2=36.5(35 + 38)/2 = 36.5(35+38)/2=36.5

Xét ngưỡng **Age ≤ 30.5**:

Age ≤ 30.5: 2 mẫu, Survived = [0, 1]:

$$G(Age \leq 30.5) = 1 - \left(\frac{1}{2}\right)^2 - \left(\frac{1}{2}\right)^2 = 1 - 0.25 - 0.25 = 0.5$$

Age > 30.5: 3 mẫu, Survived = [1, 1, 0]:

$$G(Age \leq 30.5) = 1 - \left(\frac{2}{3}\right)^2 - \left(\frac{1}{3}\right)^2 = 1 - \frac{4}{9} - \frac{1}{9} = \frac{4}{9} \approx 0.444$$

Tính Information Gain:

$$IG(Age) = 0.48 - \left(\frac{2}{5}.(0.5) + \frac{3}{5}.(0.444)\right) = 0.48 - (0.2 + 0.2664) = 0.0136$$

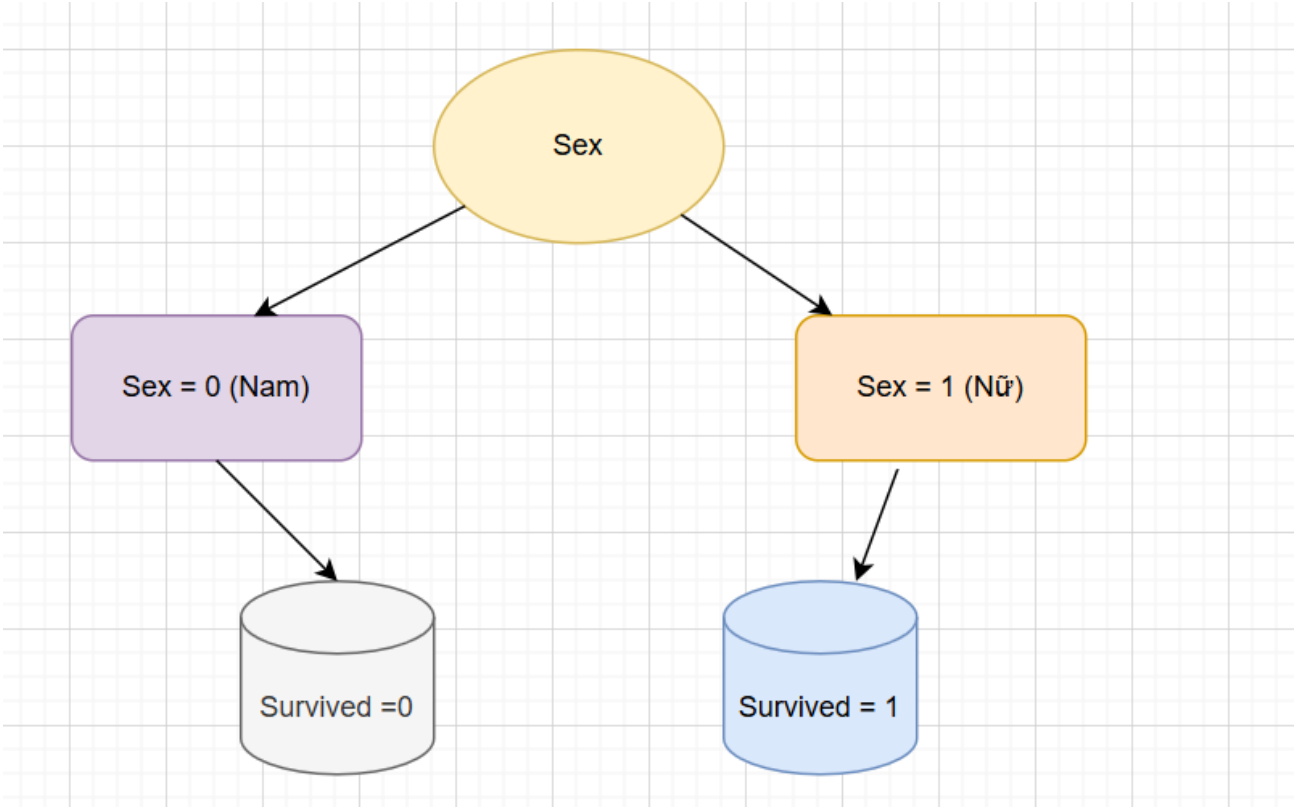
Table 12 IG của từng thuộc tính

Thuộc tính	Information Gain
Sex	0.48
Pclass	0.2136
Age	0.0136

Chọn Sex làm thuộc tính chia tốt nhất.

Xây dựng cây quyết định:

Cây (vì các nhánh đều thuần khiết nên dừng tại đây):



Hình 3.4 Xây dựng cây

Dữ liệu mới: `Pclass = 3`, `Sex = 0`, `Age = 34.5`

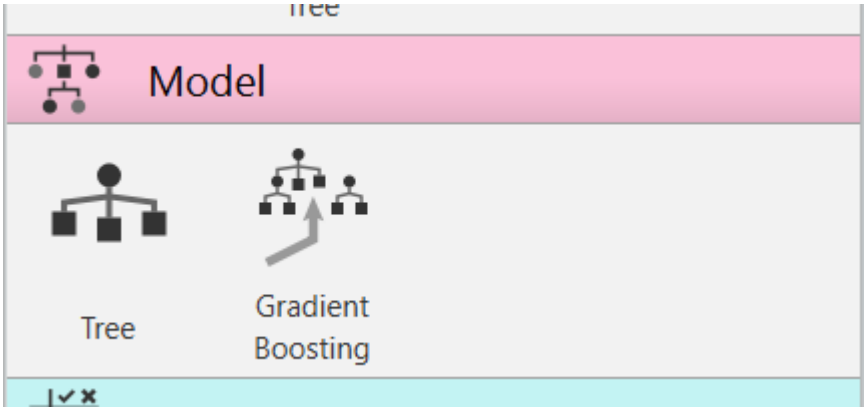
Dựa theo cây, Sex = 0 ⇒ đi nhánh trái

Dự đoán: **Survived = 0**

=> **Không sống sót**

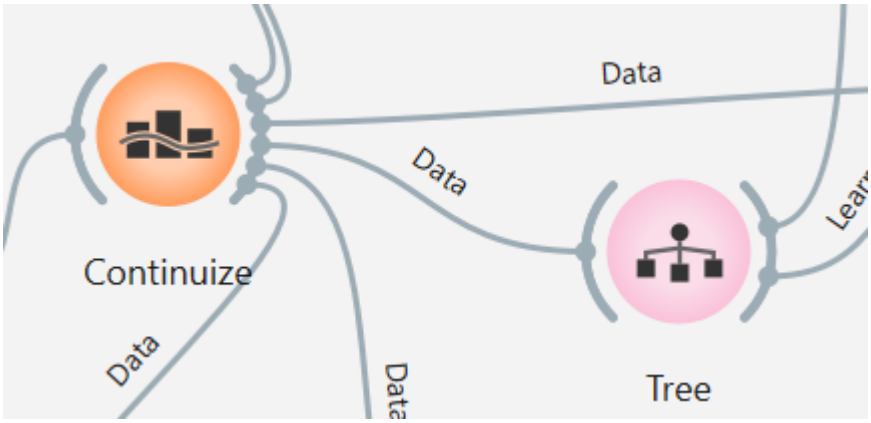
Thực hiện trên Orange:

Kéo và thả **widget Decision Tree** từ nhóm "Model" vào canvas.



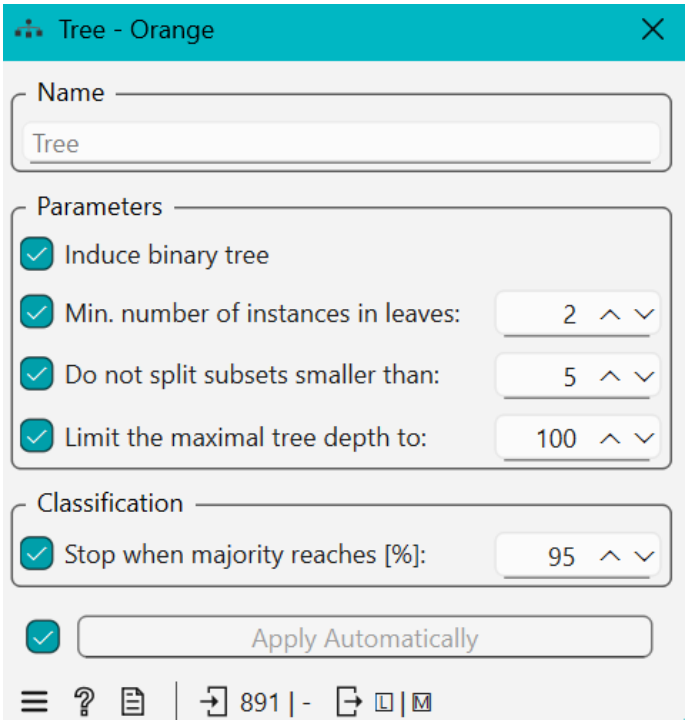
Hình 3.5 Widget

Nối đầu ra dữ liệu đã tiền xử lý đến đầu vào của **widget Decision Tree**.



Hình 3.6 Nối widget

Nhấp đúp vào widget Decision Tree để cấu hình các tham số:



Hình 3.7 Widget Decision tree

Table 13 Giải thích tham số

Tham số	Giá trị thiết lập	Giải thích ngắn
Induce binary tree	Tích chọn vào	Chia nhị phân mỗi nút
Min. number of instances in leaves	2	Đặt là **2**. Điều này quy định rằng mỗi nút lá (cuối cùng của cây) phải chứa ít nhất 2 mẫu dữ liệu. Nếu một nút được chia mà kết quả một nhánh có ít hơn 2 mẫu, việc chia đó có thể bị dừng lại để tránh overfitting.
Do not split subsets smaller than	5	Đặt là **5**. Nếu một tập con dữ liệu (tại một nút) có số lượng

		mẫu nhỏ hơn 5, cây sẽ không tiếp tục chia nút đó nữa. Điều này cũng giúp kiểm soát độ sâu của cây và ngăn chặn overfitting.
Limit the maximal tree depth to	100	Đặt là **100**. Đây là giới hạn độ sâu tối đa của cây. Mặc dù 100 là một giá trị lớn và thường không đạt tới với các tập dữ liệu nhỏ, nó giúp ngăn chặn cây phát triển vô hạn trong một số trường hợp.
Stop when majority reaches [%]	95	Đặt là **95**. Cây sẽ dừng việc chia một nút nếu 95% số mẫu trong nút đó đã thuộc cùng một lớp (tức là nút đã đủ "thuần khiết"). Điều này cũng giúp kiểm soát overfitting.

3.3. Mô hình Naive Bayes

Naïve Bayes là một thuật toán học máy **đơn giản nhưng mạnh mẽ**, thuộc nhóm học có giám sát và dựa trên **lý thuyết xác suất Bayes**. Nó được sử dụng chủ yếu cho các bài toán phân loại và đặc biệt **phù hợp với dữ liệu có phân phối chuẩn** cũng như thể hiện **tốc độ xử lý nhanh**.

Thực nghiệm thủ công:

Với dữ liệu train vẫn là 5 mẫu đầu tiên và tập test là

Table 14 Dữ liệu train

Pclass	Sex	Age
3	0	34.5

Tính toán từng phần

Công thức Bayes: $P(Y \mid X|) \infty P(Y) \cdot \prod_{i=1}^{n} P(X_i \mid Y|)$

Ta sẽ tính:

P(Survived = 0 |X)

P(Survived = 1| X)

Sau đó chọn nhãn có xác xuất lớn hơn.

Xác suất tiên nghiệm (Prior probabilities):

Từ 5 mẫu:

Sống sót = 1: 3 mẫu -> $P(1) = \frac{3}{5} = 0.6$

Không sống sót = 0: 2 mẫu ->$P(0) = \frac{2}{5} = 0.4$

Phân phối thuộc tính liên tục – Age (theo phân phối chuẩn):

Giả sử Age theo **phân phối chuẩn (Gaussian)** với công thức:

$$P(x \mid Y|) = \frac{1}{\sqrt{2\pi\sigma x^2}} \varepsilon^{\frac{(x-\mu)^2}{2\sigma^2}}$$

Với *Y=0* (mẫu 0, 4):

Age = 22.0, 35.0

$$\mu_0 = \frac{22+35}{2} = 28.5$$

$$\sigma_0^2 = \frac{(22-28.5)^2 + (35-28.5)^2}{2} = \frac{42.25 + 42.25}{2} = 42.25$$

Với *Y=1* (mẫu 1, 2, 3):

Age = 38.0, 26.0, 35.0

$$\mu_1 = \frac{38+26+35}{3} = \frac{99}{3} = 33.0$$

$$\sigma_0^2 = \frac{(38-33)^2 + (26-33)^2 + (35-33)^2}{3} = \frac{25+49+4}{3}$$

$$= \frac{78}{3} = 26.0$$

Tính xác suất có điều kiện với mẫu test: Age = 34.5

$$P(Age=34.5|Y=0) \approx \frac{1}{\sqrt{2\pi \cdot 42.25}} \cdot \varepsilon - \frac{(34.5-28.5)^2}{2 \cdot 42.25}$$

$$= \frac{1}{\sqrt{265.6}} \cdot \varepsilon - \frac{36}{84.2} \approx \frac{1}{16.3} \cdot \varepsilon^{-0.426} \approx 0.061 \cdot 0.653 = 0.0398$$

$$P(Age=34.5|Y=1) \approx \frac{1}{\sqrt{2\pi \cdot 26}} \cdot \varepsilon - \frac{(34.5-33)^2}{2 \cdot 26}$$

$$= \frac{1}{\sqrt{163.36}} \cdot \varepsilon - \frac{2.25}{52} \approx \frac{1}{12.8} \cdot \varepsilon^{-0.043} \approx 0.078 \cdot 0.958 = 0.0747$$

Tính xác suất hậu nghiệm:

$$P(Y=0 \mid X)$$

$$\propto P(0) \cdot P(Pclass=3 \mid 0) \cdot P(Sex=0 \mid 0) \cdot P(Age=34.5 \mid 0)$$

$$= 0.4 \cdot 1.0 \cdot 1.0 \cdot 0.0398 = 0.0159$$

$$P(Y=1 \mid X)$$

$$\propto P(1) \cdot P(Pclass=3 \mid 1) \cdot P(Sex=0 \mid 1)$$

$$\cdot P(Age=34.5 \mid 1)$$

$P(Sex=0 \mid 1)=0 \rightarrow$ sẽ làm xác xuất bị 0 => dùng **Laplace smoothing**:

Giả sử ta có 2 giá trị cho Sex ⇒ áp dụng Laplace:

$$P(Sex=0 \mid Y=1) = \frac{0+1}{3+2} = \frac{1}{5} = 0.2$$

$$= 0.6 \cdot \frac{1}{3} \cdot 0.2 \cdot 0.0747 = 0.6 \cdot 0.333 \cdot 0.2 \cdot 0.0747 \approx 0.00299$$

So sánh & Dự đoán:

Table 15 So sánh

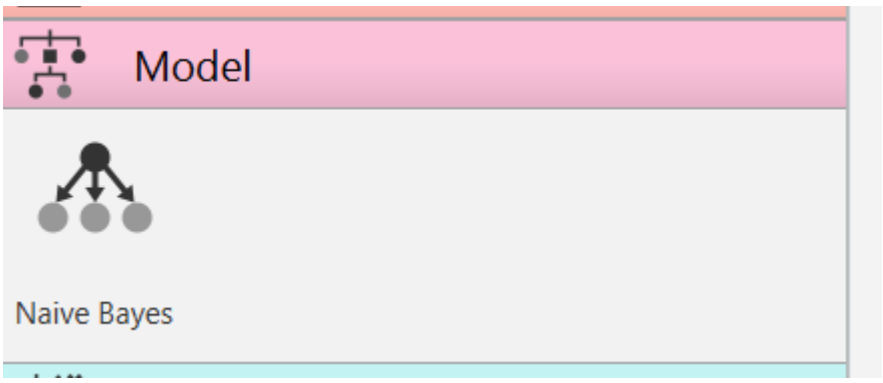
Nhãn	Xác suất hậu nghiệm
0 (không sống sót)	0.0159
1 (sống sót)	0.00299

Dự đoán cuối cùng: Survived = 0 (Không sống sót)

Thực nghiệm trên công cụ Orange:

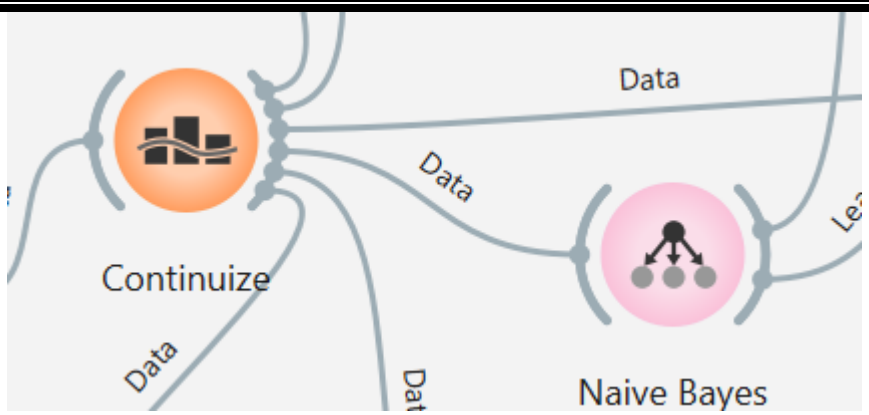
Các bước chọn dữ liệu, chọn đặc trưng và mã hóa dữ liệu đều giống như các bước đã thực hiện cho kNN và Decision Tree.

Kéo và thả **widget Naive Bayes** từ nhóm "Model" vào canvas.



Hình 3.8 widget

Nối đầu ra dữ liệu đã tiền xử lý đến đầu vào của **widget Naive Bayes**.



Hình 3.9 Nối widget

Nhấp đúp vào widget Naive Bayes:

Thiết lập tham số:

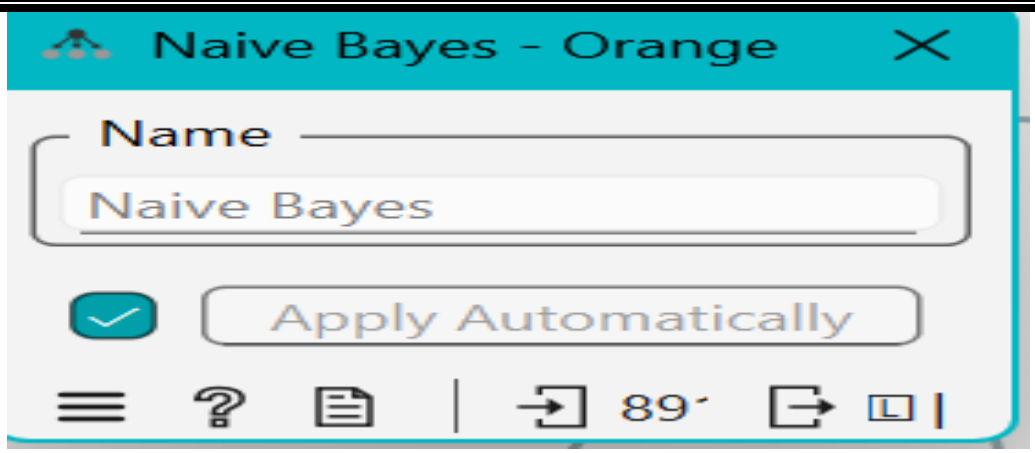
Trong Orange, Naive Bayes mặc định **không có nhiều tham số để chỉnh** vì nó là mô hình đơn giản. Tuy nhiên, bạn nên kiểm tra:

Apply Automatically: Bật (để cập nhật tự động)

Tên mô hình: Có thể đặt lại tên là Naive Bayes hoặc NB

Không cần thiết lập gì thêm, vì:

- Orange dùng Gaussian Naive Bayes (với biến liên tục như Age)
- Các biến phân loại (như Sex) được xử lý tự động
- Laplace smoothing được tích hợp sẵn trong thư viện scikit-learn backend của Orange, giúp tránh xác suất bằng 0 khi một tổ hợp thuộc tính/lớp không xuất hiện trong dữ liệu huấn luyện.



Hình 3.10 Widget Naïve Bayes

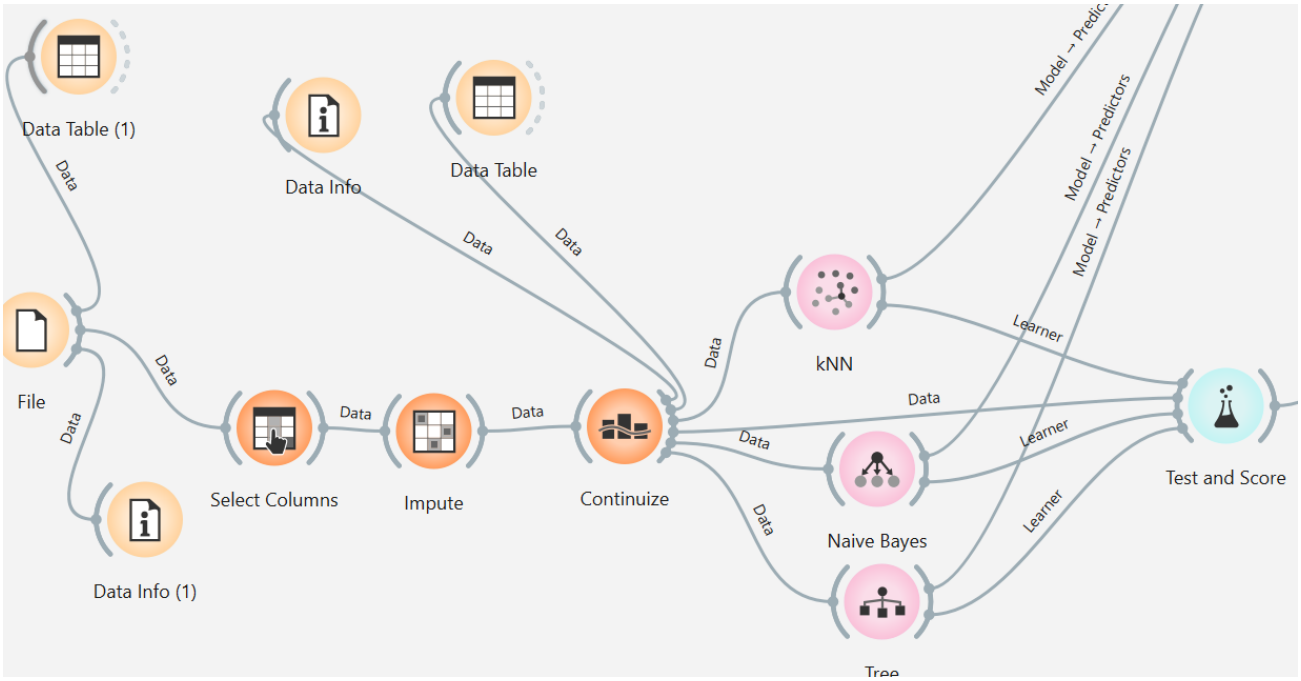
3.4. Thước đo đánh giá mô hình

3.4.1. Thực hiện các bước để đánh giá trên Orange:

Bước 1: Thước đo đánh giá Test and Score:

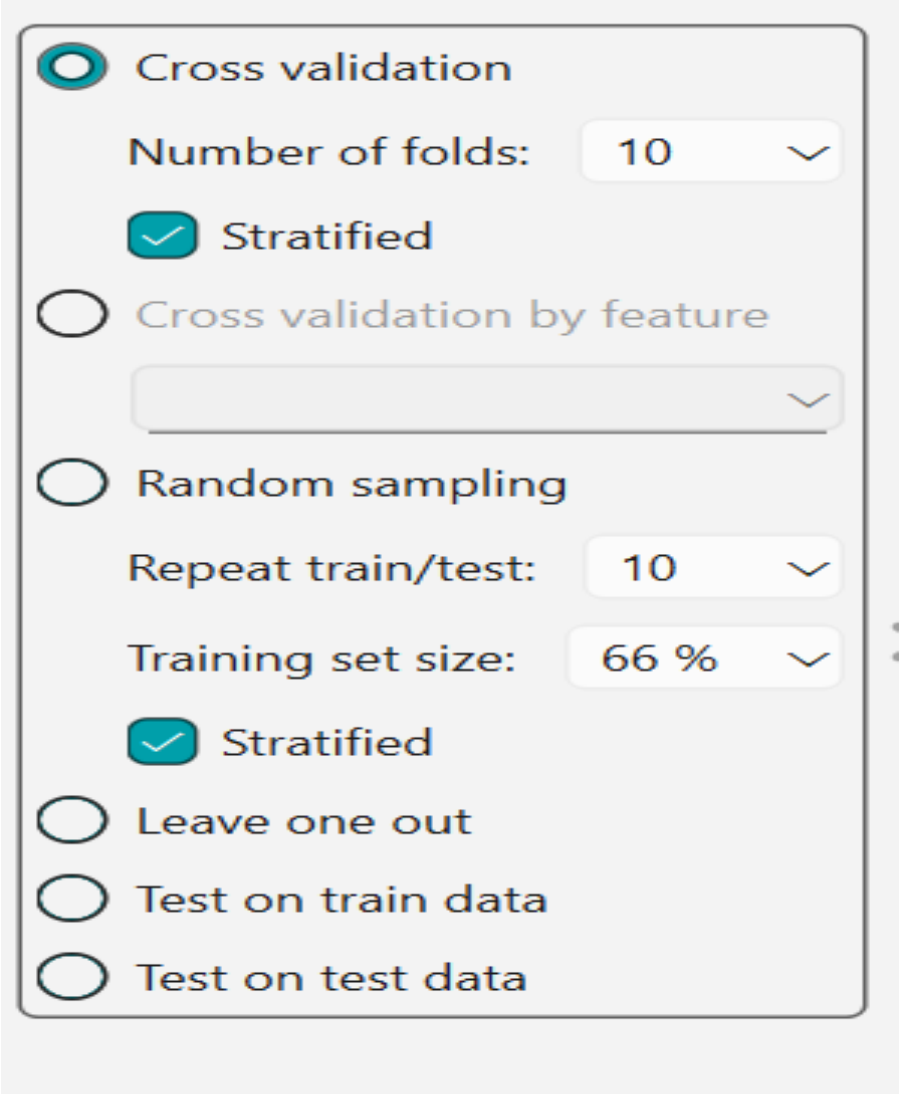
Kéo widget **Test and Score** vào canvas.

Nối **Learner** từ mô hình (kNN, Decision Tree, Naive Bayes) và **Data** từ widget **Continuize** (sau khi đã tiền xử lý dữ liệu) vào Test and Score.



Hình 3.11 Đánh giá test and Score

Trong Test and Score, chọn **Evaluation Method: Cross Validation** và **Number of folds: 10**. Đảm bảo **Stratified** được chọn.



Hình 3.12 Đánh giá test and socre

Evaluation Method: Cross Validation: Đây là phương pháp đánh giá mô hình được chọn. Cross-validation (kiểm định chéo) là một kỹ thuật phổ biến để đánh giá hiệu suất của mô hình học máy một cách đáng tin cậy hơn so với việc chỉ chia tập dữ liệu thành tập huấn luyện và tập kiểm tra một lần duy nhất. Nó giúp đảm bảo rằng mô hình không bị quá khớp (overfitting) với một tập con dữ liệu cụ thể và có khả năng tổng quát hóa tốt hơn trên dữ liệu chưa thấy.

Number of folds: 10: Đây là số lượng "folds" (lần chia) trong quá trình Cross-validation. Khi bạn chọn 10 folds, tập dữ liệu sẽ được chia thành 10 phần bằng nhau. Mô hình sẽ được

huấn luyện 10 lần; mỗi lần, một phần khác nhau sẽ được dùng làm tập kiểm tra và 9 phần còn lại được dùng làm tập huấn luyện. Kết quả cuối cùng là giá trị trung bình của hiệu suất trên cả 10 lần chạy, giúp đánh giá mô hình một cách khách quan hơn.

Check Stratified: Khi tùy chọn "Stratified" được tích chọn, quá trình chia dữ liệu thành các folds sẽ đảm bảo rằng tỷ lệ các lớp (ví dụ: lớp "sống sót" và "không sống sót" trong bài toán Titanic) được duy trì tương tự nhau trong mỗi fold. Điều này rất quan trọng đối với các tập dữ liệu có sự mất cân bằng giữa các lớp, vì nó giúp tránh trường hợp một fold nào đó chỉ chứa toàn bộ mẫu của một lớp, dẫn đến đánh giá hiệu suất không chính xác cho mô hình

Random Sampling là một phương pháp đánh giá mô hình thay thế cho Cross Validation, và nó **không được tích chọn** trong báo cáo:

Mục đích: Khi được chọn, Random Sampling sẽ chia tập dữ liệu gốc thành hai phần: một tập huấn luyện (training set) và một tập kiểm tra (test set) một cách ngẫu nhiên

Training Set Size (ví dụ: 80%): Xác định tỷ lệ phần trăm dữ liệu sẽ được sử dụng để huấn luyện mô hình. Phần còn lại (ví dụ: 20%) sẽ được dùng làm tập kiểm tra

Repeat (ví dụ: 10 times): Xác định số lần quá trình chia ngẫu nhiên và đánh giá được lặp lại. Kết quả cuối cùng sẽ là trung bình của các lần lặp này

So với **Cross Validation**, **Random Sampling** thường được sử dụng khi kích thước dữ liệu rất lớn hoặc khi cần kiểm tra hiệu suất mô hình một cách nhanh chóng. Tuy nhiên, Cross Validation thường được ưu tiên vì nó sử dụng toàn bộ dữ liệu để huấn luyện và kiểm tra một cách có hệ thống hơn, giúp đánh giá mô hình đáng tin cậy hơn và giảm thiểu sự thiên vị.

Test on training data:

Nếu được chọn, mô hình sẽ được huấn luyện và sau đó kiểm tra ngay trên chính tập dữ liệu mà nó đã được huấn luyện

Ưu và nhược điểm: Mặc dù cho kết quả độ chính xác rất cao, phương pháp này **không thể hiện khả năng tổng quát hóa** của mô hình trên dữ liệu mới. Độ chính xác trên tập huấn luyện thường cao một cách giả tạo và không phản ánh hiệu suất

thực tế khi mô hình gặp dữ liệu chưa từng thấy. Do đó, nó hiếm khi được sử dụng để đánh giá cuối cùng về hiệu suất mô hình trong thực tế.

Test on separate test data:

Khi được chọn, bạn sẽ cung cấp một tập dữ liệu kiểm tra riêng biệt, hoàn toàn độc lập với tập dữ liệu huấn luyện.

Phương pháp này rất hữu ích khi bạn đã có một tập dữ liệu đã được phân tách rõ ràng thành tập huấn luyện và tập kiểm tra (ví dụ: từ các cuộc thi Kaggle nơi có tập train.csv và test.csv riêng biệt cho việc dự đoán cuối cùng). Tuy nhiên, trong ngữ cảnh đánh giá hiệu suất nội bộ hoặc khi không có tập kiểm tra độc lập, Cross Validation thường được ưu tiên.

Bước 2: Đánh giá bằng Confusion Matrix

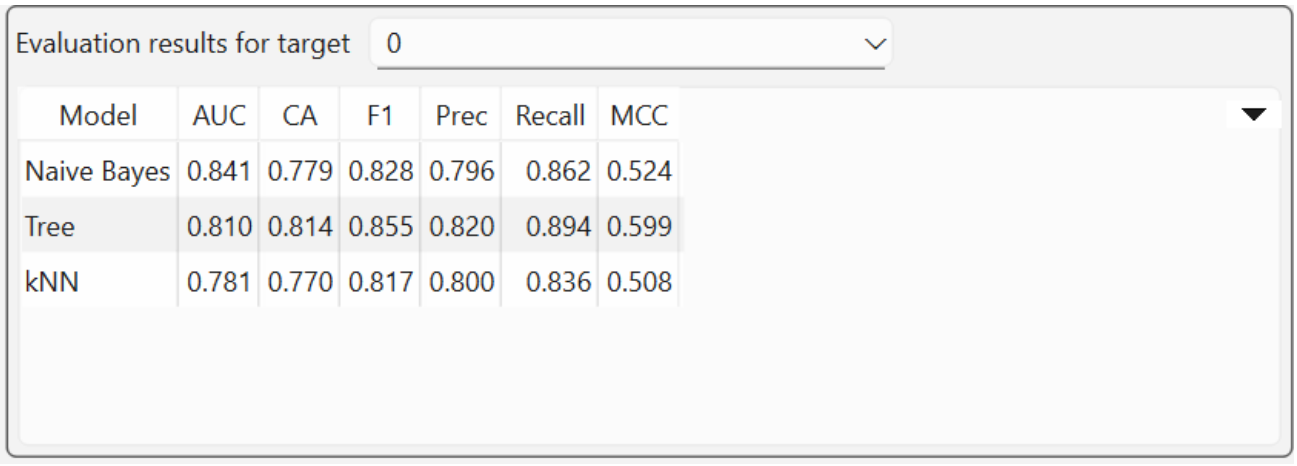
Kéo thêm widget **Confusion Matrix** vào canvas và nối nó với Test and Score.

Trong Confusion Matrix, có thể quan sát số lượng mẫu được dự đoán đúng (TP, TN) và sai (FP, FN) cho từng lớp (0 - không sống sót và 1 - sống sót).

3.4.2 Đánh giá bằng thước đo

Test and Score:

a) Lớp mục tiêu = 0 (Không sống sót):



Model	AUC	CA	F1	Prec	Recall	MCC
Naive Bayes	0.841	0.779	0.828	0.796	0.862	0.524
Tree	0.810	0.814	0.855	0.820	0.894	0.599
kNN	0.781	0.770	0.817	0.800	0.836	0.508

Hình 3.13 Đánh giá so sánh

Mô hình **Tree** đạt **CA (Classification Accuracy) cao nhất = 0.814** và **Recall cao nhất = 0.894**.

CA (Classification Accuracy): là tỷ lệ tổng số mẫu được dự đoán đúng trên tổng số mẫu. CA = 0.814 nghĩa là mô hình Tree đoán đúng **81.4% tổng số trường hợp** trong toàn bộ tập dữ liệu, cho thấy hiệu suất tổng thể rất tốt.

Recall (Độ nhạy) cho lớp 0 là 0.894: Cho biết mô hình đã phát hiện được **89.4% số người thực sự không sống sót**, tức là khả năng nhận diện đúng người **không sống sót** là rất cao – điều này đặc biệt quan trọng khi ta quan tâm đến việc không bỏ sót các trường hợp nguy hiểm.

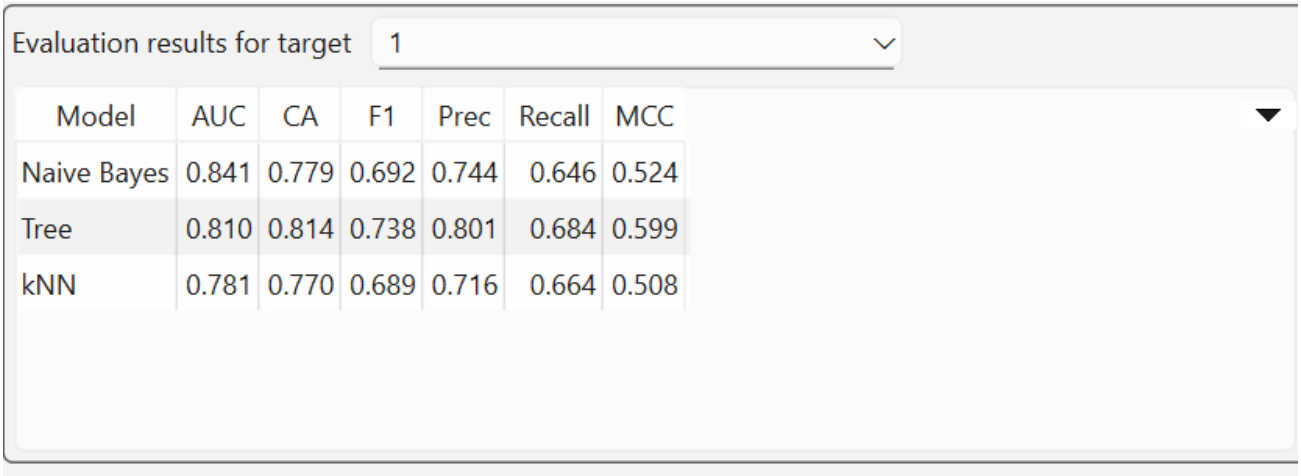
Mô hình **Tree** cũng đạt **F1-score = 0.855** và **MCC = 0.599** cao nhất trong 3 mô hình.

F1-score là trung bình điều hòa giữa **Precision** và **Recall**, dùng để đánh giá hiệu suất mô hình trong các trường hợp dữ liệu mất cân bằng. F1 = 0.855 cho thấy **sự cân bằng rất tốt** giữa việc phát hiện đúng người không sống sót và không dự đoán sai người sống sót.
MCC (Matthews Correlation Coefficient) là chỉ số đánh giá hiệu quả mô hình dựa trên tất cả bốn giá trị trong Confusion Matrix (TP, TN, FP, FN). MCC = 0.599 cho thấy **mối tương quan mạnh mẽ giữa dự đoán và thực tế**, phản ánh hiệu suất mô hình một cách cân bằng và toàn diện.

Mô hình **Naive Bayes** có **AUC = 0.841** (**cao nhất**) → đo lường **khả năng phân biệt** giữa hai lớp. AUC gần 1 thể hiện mô hình phân loại tốt. Tuy nhiên, các chỉ số khác như **CA = 0.779**, **F1 = 0.828**, **MCC = 0.524** đều **thấp hơn Tree**, cho thấy hiệu suất tổng thể kém hơn.

Mô hình **kNN** có **F1 = 0.817**, **Recall = 0.836**, **MCC = 0.508** → là mô hình có hiệu suất yếu nhất cho lớp không sống sót, thể hiện ở các chỉ số đều thấp nhất trong ba mô hình.

b) Lớp mục tiêu = 1 (Sống sót)



Model	AUC	CA	F1	Prec	Recall	MCC
Naive Bayes	0.841	0.779	0.692	0.744	0.646	0.524
Tree	0.810	0.814	0.738	0.801	0.684	0.599
kNN	0.781	0.770	0.689	0.716	0.664	0.508

Hình 3.14 Đánh giá so sánh

Decision Tree tiếp tục thể hiện hiệu quả vượt trội với F1-score (0.738) và MCC (0.599) cao nhất. Điều này củng cố nhận xét rằng Tree có hiệu suất tổng thể tốt, không chỉ với lớp 0 mà còn với lớp 1, đạt được sự cân bằng giữa Precision và Recall.

Decision Tree có **Precision cao nhất (0.801)** và **Recall cao nhất (0.684)**.

Precision (Độ chính xác) cho lớp 1 là 0.801: Điều này có nghĩa là trong số tất cả những người mà Decision Tree dự đoán là "sống sót", có tới **80.1%** thực sự sống sót. Mô hình này **khá chắc chắn** khi dự đoán một người sống sót.
Recall cho lớp 1 là 0.684: Cho thấy Decision Tree đã xác định đúng **68.4% số người thực sự sống sót**. Mặc dù chính xác khi dự đoán, mô hình vẫn **bỏ sót khoảng 31.6%** trường hợp sống sót, tuy nhiên đây **vẫn là chỉ số Recall cao nhất** trong 3 mô hình.

Naive Bayes có **AUC = 0.841** cao nhất nhưng các chỉ số còn lại đều thấp hơn Tree, như F1-score (0.692), Recall (0.646), MCC (0.524) → cho thấy hiệu suất tổng thể vẫn kém hơn **Tree**.

kNN có các chỉ số **F1-score = 0.689**, **Recall = 0.664**, **MCC = 0.508** thấp nhất trong ba mô hình → cho thấy mô hình này hoạt động kém nhất ở lớp sống sót.

Kết luận từ Test and Score:

Decision Tree là mô hình **ổn định và hiệu quả nhất** trong cả hai trường hợp (sống sót và không sống sót). Với **F1-score, Accuracy và MCC cao nhất**, đây là mô hình phù hợp nhất để áp dụng trong việc dự đoán sống sót dựa trên dữ liệu Titanic.

Naive Bayes chỉ nổi bật ở **AUC**, thể hiện khả năng phân biệt giữa các lớp, nhưng lại **thiếu sự cân bằng và hiệu quả thực tế**, khiến nó không phải là lựa chọn tối ưu.

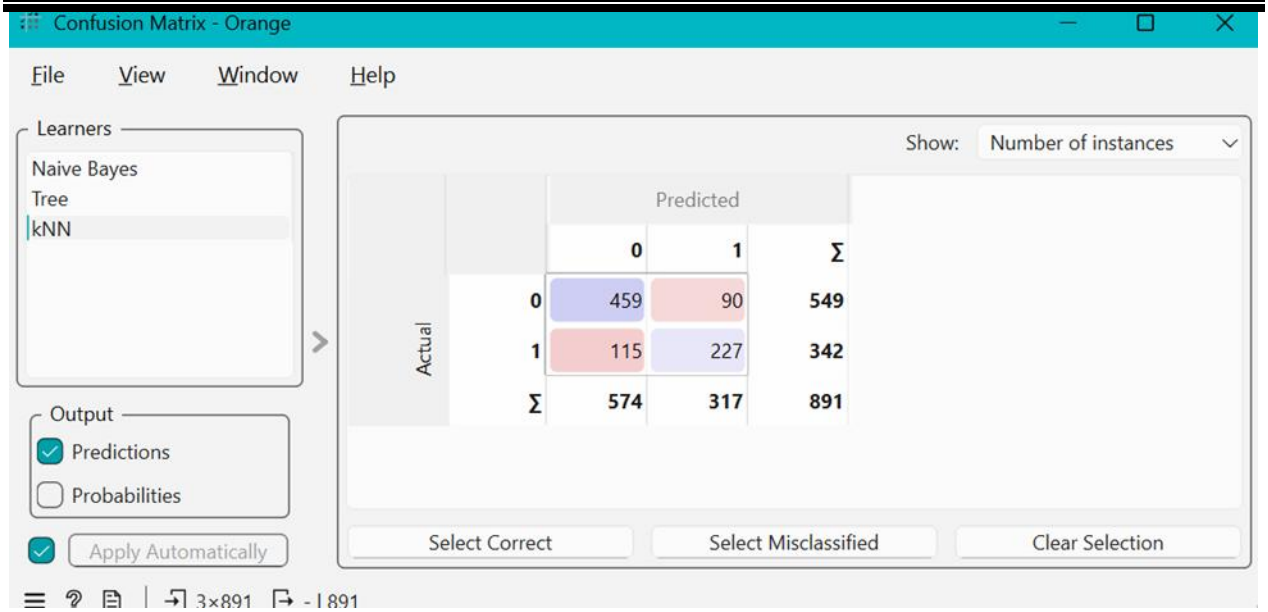
kNN không đạt hiệu suất nổi bật ở bất kỳ chỉ số nào, cho thấy **mô hình này không phù hợp với dữ liệu và bài toán hiện tại**.

Confusion Matrix:

Confusion Matrix cung cấp cái nhìn chi tiết hơn về các loại lỗi mà mô hình mắc phải (dự đoán đúng, dự đoán sai). Các giá trị trong ma trận được giải thích như sau:

- **TP (True Positive)**: Số lượng mẫu của lớp dương (ở đây là "sống sót" = 1) được dự đoán đúng là lớp dương.
- **TN (True Negative)**: Số lượng mẫu của lớp âm (ở đây là "không sống sót" = 0) được dự đoán đúng là lớp âm.
- **FP (False Positive)**: Số lượng mẫu của lớp âm được dự đoán sai là lớp dương (lỗi Type I).
- **FN (False Negative)**: Số lượng mẫu của lớp dương được dự đoán sai là lớp âm (lỗi Type II).

Mô hình kNN



Hình 3.15 Confusion Matrix KNN

TP (True Positive) = 227

→ Hành khách **thực sự sống sót** và **được dự đoán đúng** là sống sót.

FP (False Positive) = 90

→ Hành khách **không sống sót** nhưng **dự đoán sai** là sống sót.

FN (False Negative) = 115

→ Hành khách **sống sót** nhưng **dự đoán sai** là không sống sót.

TN (True Negative) = 459

→ Hành khách **không sống sót** và **được dự đoán đúng** là không sống sót.

Precision (lớp 1) ≈ 0.716: (227 / (227 + 90)). Trong số những người mà mô hình **dự đoán là sống sót**, có **71.6% thực sự sống sót**.

Recall (lớp 1) ≈ 0.664: (227 / (227 + 115)). Mô hình đã **tìm được 66.4% số hành khách thực sự sống sót**.

F1-score (lớp 1) ≈ 0.689.

Precision (Lớp 0**)** ≈ 0.799: (459 / (459 + 115)). Trong số những hành khách mà mô hình **dự đoán là không sống sót**, có **79.9% thực sự không sống sót**.

Recall (lớp 0) ≈ 0.836: (459 / (459 + 90)). Mô hình đã **nhận diện đúng 83.6% số hành khách thực sự không sống sót**.

F1-score (lớp 0) ≈ 0.817.

Nhận xét

Mô hình **kNN phân biệt khá tốt** nhóm hành khách không sống sót.

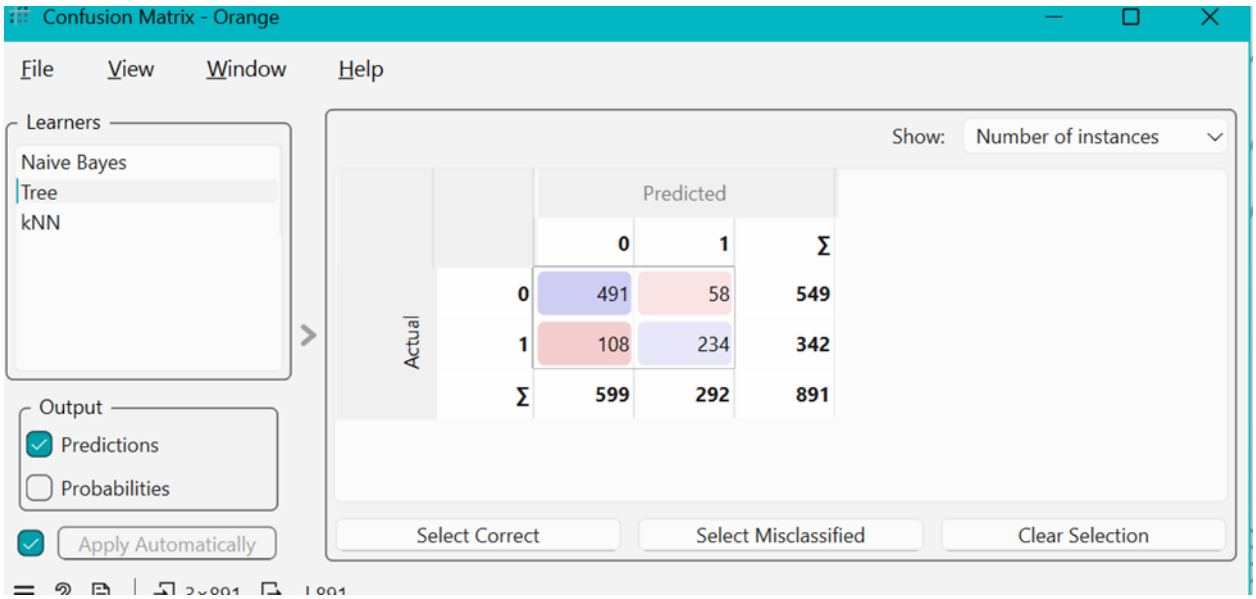
F1-score của lớp 0 (≈ **0.817**) **tốt hơn đáng kể** so với lớp 1 (≈ 0.689).

Có độ chính xác tương đối khá trong việc nhận diện hành khách sống sót, nhưng vẫn còn bỏ sót khá nhiều người thực sự sống sót (Recall chưa cao).

Precision (71.6%) cao hơn Recall (66.4%) cho thấy mô hình **có xu hướng cẩn trọng**, chỉ dự đoán sống sót khi khá chắc chắn.

F1-score (≈ 0.689) cho thấy mức độ cân bằng giữa Precision và Recall **chưa quá lý tưởng**, có thể cải thiện thêm.

Mô hình Decision Tree:



Hình 3.16 Confusion Matrix Decision Tree

Lớp 1:

TP = 234:

→ Có 234 hành khách thực sự sống sót và được Decision Tree dự đoán đúng là sống sót.

FP = 58

→ Có 58 hành khách thực sự không sống sót nhưng Decision Tree lại dự đoán sai là sống sót.

FN = 108

→ Có 108 hành khách thực sự sống sót nhưng Decision Tree lại dự đoán sai là không sống sót.

TN = 491

→ Có 491 hành khách thực sự không sống sót và được Decision Tree dự đoán đúng là không sống sót.

Precision (lớp 1) ≈ 0.801: (234 / (234 + 58)). Decision Tree rất chính xác khi dự đoán "sống sót".

Recall (lớp 1) ≈ 0.684: (234 / (234 + 108)). Tuy nhiên, nó bỏ sót một phần người sống sót.

F1-score (lớp 1) ≈ 0.738.

Lớp 0:

TP = 491

→ Có 491 hành khách thực sự **không sống sót** và được Decision Tree **dự đoán đúng** là không sống sót.

FP = 108

→ Có 108 hành khách thực sự **sống sót** nhưng bị Decision Tree **dự đoán sai** là không sống sót.

FN = 58

→ Có 58 hành khách thực sự **không sống sót** nhưng bị Decision Tree **dự đoán sai** là sống sót.

TN = 234

→ Có 234 hành khách thực sự **sống sót** và được Decision Tree **dự đoán đúng** là sống sót.

Precision (lớp 0) ≈ 0.894: (491 / (491 + 58)). Decision Tree rất chính xác khi dự đoán "không sống sót".

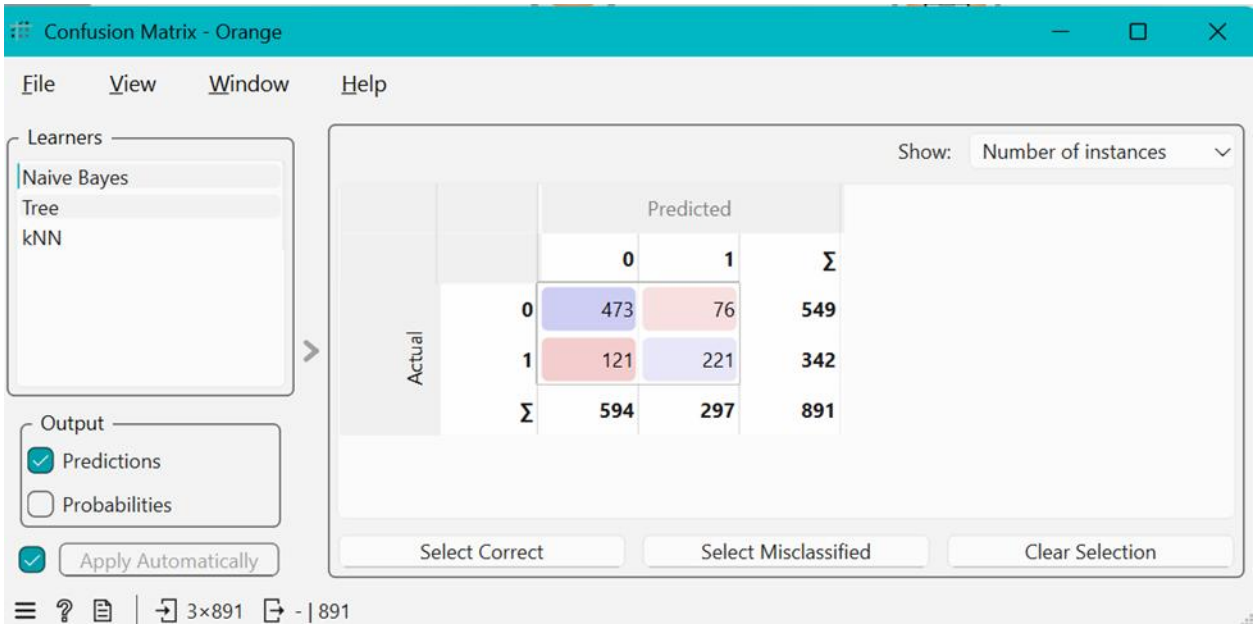
Recall (lớp 0) ≈ 0.684: (491 / (491 + 108)). Phần lớn người không sống sót được mô hình phát hiện đúng, chỉ bỏ sót một phần.

F1-score (lớp 0) ≈ 0.855.

Nhận xét:

- Decision Tree nghiêng về việc phát hiện "không sống sót" chính xác hơn là "sống sót".
- Với **Precision lớp 1 cao** nhưng **Recall thấp**, mô hình **cẩn trọng khi dự đoán sống sót** – chỉ dự đoán khi chắc chắn.
- Trong khi đó, **lớp 0 được dự đoán đều hơn về Precision và Recall**, cho thấy độ tin cậy cao khi phân loại người không sống sót.
- Nếu mục tiêu là **hạn chế bỏ sót người sống sót** (Recall lớp 1), thì cần cải thiện thêm mô hình này.
- Còn nếu mục tiêu là **dự đoán chính xác người không sống sót**, mô hình hiện tại đang làm tốt.

Mô hình Naive Bayes:



Hình 3.17 Confusion Matrix Naïve Bayes

Lớp 1 (Sống sót):

TP = 221: Có 221 hành khách thực sự sống sót và được Naive Bayes dự đoán đúng là sống sót.

FP = 76: Có 76 hành khách thực sự không sống sót nhưng Naive Bayes lại dự đoán sai là sống sót.

FN = 121: Có 121 hành khách thực sự sống sót nhưng Naive Bayes lại dự đoán sai là không sống sót.

TN = 473: Có 473 hành khách thực sự không sống sót và được Naive Bayes dự đoán đúng là không sống sót.

Precision (lớp 1) ≈ 0.744: (221 / (221 + 76)).

Recall (lớp 1) ≈ 0.681: (221 / (221 + 121)).

F1-score (lớp 1) ≈ 0.691.

Lớp 0 (Không sống sót):

TP = 473: Có 473 hành khách thực sự sống sót và được Naive Bayes dự đoán đúng là sống sót.

FP = 121: Có 121 hành khách thực sự không sống sót nhưng Naive Bayes lại dự đoán sai là sống sót.

FN = 76: Có 76 hành khách thực sự sống sót nhưng Naive Bayes lại dự đoán sai là không sống sót.

TN = 221: Có 221 hành khách thực sự không sống sót và được Naive Bayes dự đoán đúng là không sống sót.

Precision (lớp 0) ≈ 0.796: (473 / (473 + 121)).

Recall (lớp 0) ≈ 0.861: (473 / (473 + 76)).

F1-score (lớp 0) ≈ 0.827.

Nhận xét:

Mô hình **Naive Bayes** hoạt động **tốt hơn ở lớp 0 (không sống sót)** với độ chính xác và bao phủ cao hơn.

Ở **lớp 1 (sống sót)**, mô hình **bỏ sót khá nhiều người sống sót (FN = 121)** và cũng có **sai sót khi dự đoán người không sống (FP = 76)**.

Điều này cho thấy **Naive Bayes thiên về dự đoán "không sống sót" hơn**, và **chưa thực sự hiệu quả với lớp "sống sót"**.

TỔNG KẾT SO SÁNH CẢ 3 MÔ HÌNH:

Table 16 Tổng kết so sánh cả 3 mô hình

Mô hình	Precision (1)	Recall (1)	F1-score (1)	Nhận xét chính
kNN	0.716	0.664	0.689	Precision và Recall khá cân bằng. Mô hình ổn định, nhưng không nổi bật ở chỉ số nào.
Tree	**0.801**	**0.684**	**0.738**	Precision cao nhất, tức ít dự đoán sai người không sống sót là sống sót (FP thấp).
Naive Bayes	0.744	0.681	0.691	Hiệu suất thấp hơn hai mô hình còn lại ở mọi chỉ số. Không phù hợp với tập dữ liệu này.

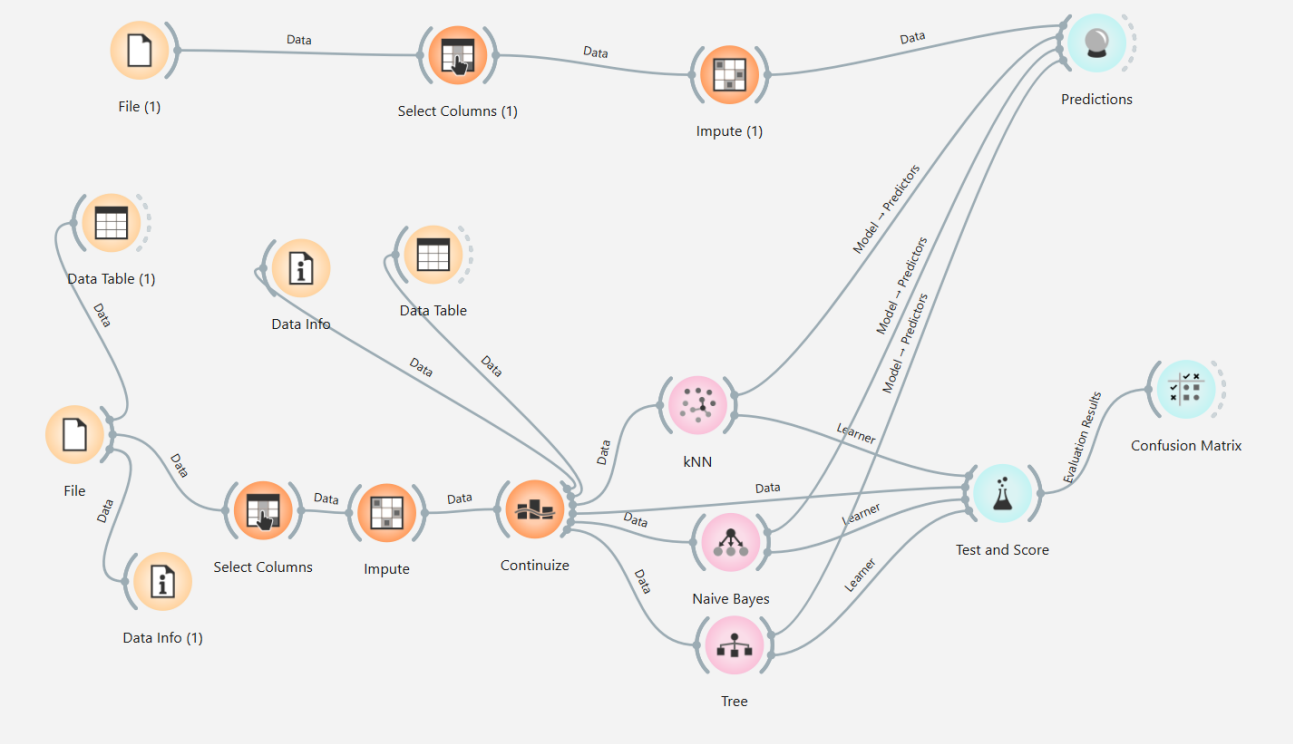
Nếu mục tiêu là đạt được sự cân bằng tốt giữa việc dự đoán đúng người sống sót (Precision) và **không bỏ sót nhiều người sống sót (Recall)**, thì **Decision Tree** là lựa chọn phù hợp nhất. Mô hình này có **Precision cao nhất (0.801)** trong khi **Recall ở mức ổn (0.684)**, giúp hạn chế cảnh báo sai và vẫn giữ được khả năng phát hiện tốt.

Nếu muốn một mô hình có độ ổn định giữa Precision và Recall mà không quá thiên lệch, thì **kNN** là lựa chọn thay thế hợp lý. Tuy các chỉ số đều thấp hơn Decision Tree, nhưng **sự cân bằng giữa Precision (0.716) và Recall (0.664)** giúp mô hình hoạt động đều và ổn định trong nhiều tình huống.

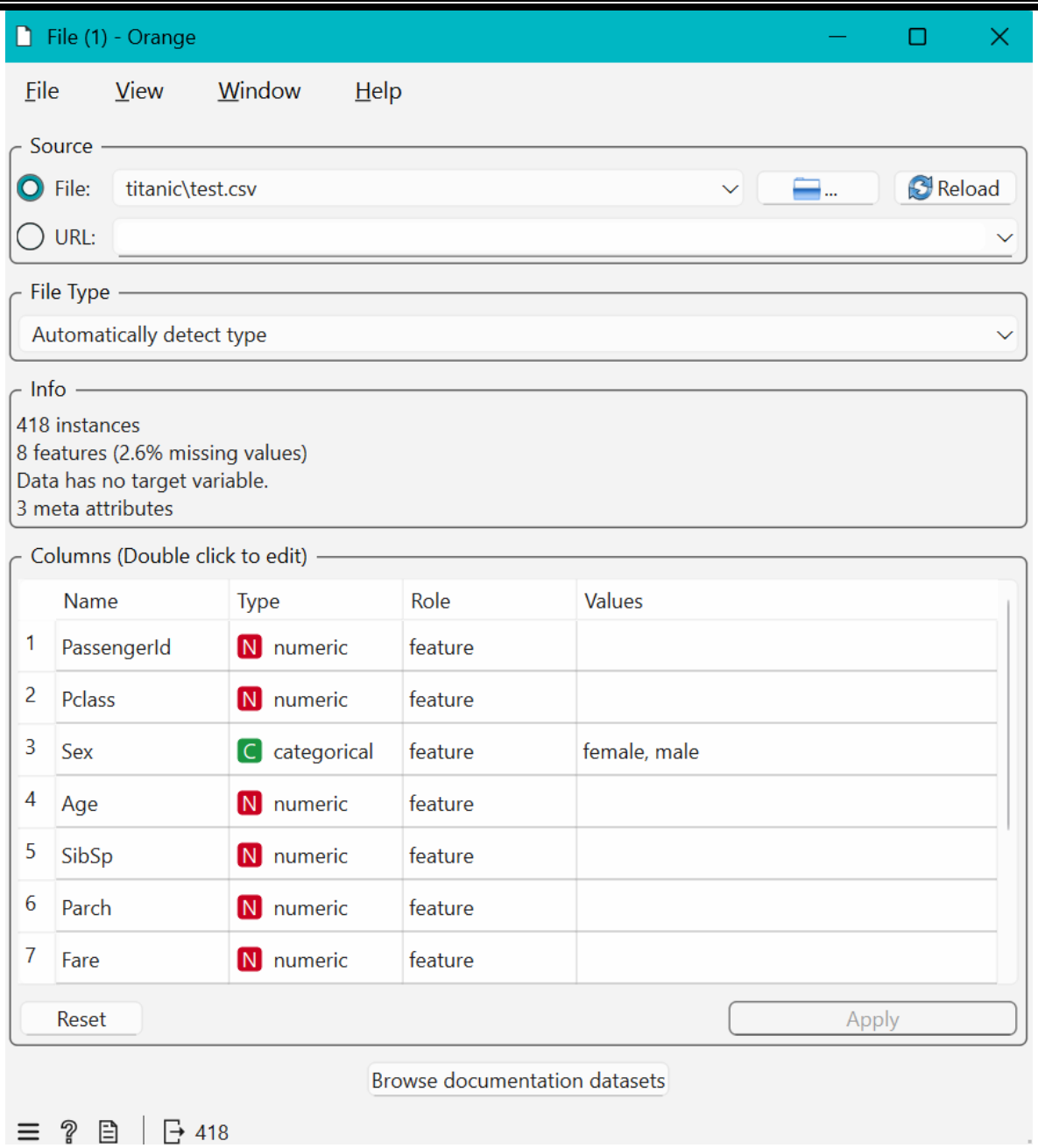
Naive Bayes cho kết quả **kém hơn cả hai mô hình còn lại** ở cả Precision (0.744) và Recall (0.681). Mô hình này **dễ bỏ sót người sống sót** và **dự đoán sai nhiều người không sống sót thành sống sót**, do đó **không phù hợp cho bài toán này**.

CHƯƠNG 4. KẾT QUẢ THỰC NGHIỆM

4.1. Pipeline xử lý bài toán dự đoán sống sót trên tàu Titanic trên Orange



Hình 4.1 Hình ảnh tổng quan về cách xử lý bài toán trong Orange

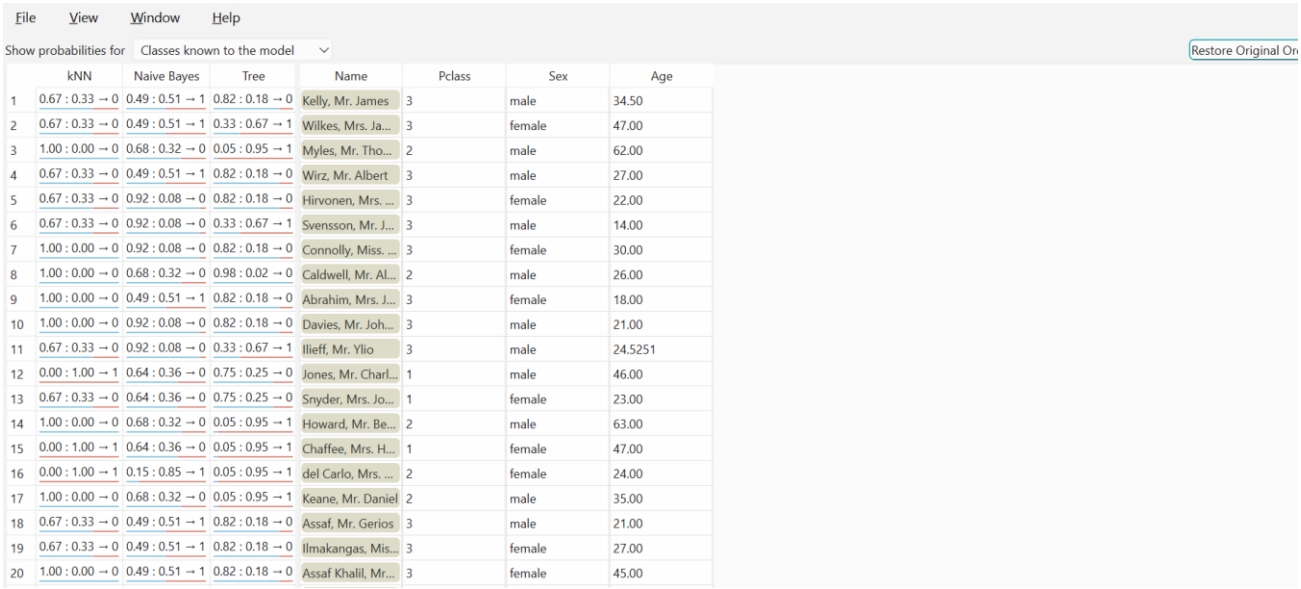


Hình 4.2 File (1) chứa dữ liệu dùng để test

Các tiền xử lí dữ liệu cho file (1) (test.csv) đều được thực thi giống như Chương 2

4.2. Kết quả dự đoán cuối cùng

Như pipline ở trên ta có thể thấy, để có thể dự đoán thì widget Predictions cần dữ liệu test và dữ liệu đã huấn luyện qua các mô hình nên chúng em nối dữ liệu đã huấn luyện từ 3 Model đến widget Predictions và cũng nối dữ liệu từ tập test đã được tiền xử lí đến Predictions tiếp.


File View Window Help

Show probabilities for Classes known to the model

Restore Original Or

	kNN	Naive Bayes	Tree	Name	Pclass	Sex	Age
1	0.67 : 0.33 → 0	0.49 : 0.51 → 1	0.82 : 0.18 → 0	Kelly, Mr. James	3	male	34.50
2	0.67 : 0.33 → 0	0.49 : 0.51 → 1	0.33 : 0.67 → 1	Wilkes, Mrs. Ja...	3	female	47.00
3	1.00 : 0.00 → 0	0.68 : 0.32 → 0	0.05 : 0.95 → 1	Myles, Mr. Tho...	2	male	62.00
4	0.67 : 0.33 → 0	0.49 : 0.51 → 1	0.82 : 0.18 → 0	Wirz, Mr. Albert	3	male	27.00
5	0.67 : 0.33 → 0	0.92 : 0.08 → 0	0.82 : 0.18 → 0	Hirvonen, Mrs. ...	3	female	22.00
6	0.67 : 0.33 → 0	0.92 : 0.08 → 0	0.33 : 0.67 → 1	Svensson, Mr. J...	3	male	14.00
7	1.00 : 0.00 → 0	0.92 : 0.08 → 0	0.82 : 0.18 → 0	Connolly, Miss. ...	3	female	30.00
8	1.00 : 0.00 → 0	0.68 : 0.32 → 0	0.98 : 0.02 → 0	Caldwell, Mr. Al...	2	male	26.00
9	1.00 : 0.00 → 0	0.49 : 0.51 → 1	0.82 : 0.18 → 0	Abrahim, Mrs. J...	3	female	18.00
10	1.00 : 0.00 → 0	0.92 : 0.08 → 0	0.82 : 0.18 → 0	Davies, Mr. Joh...	3	male	21.00
11	0.67 : 0.33 → 0	0.92 : 0.08 → 0	0.33 : 0.67 → 1	Ilieff, Mr. Ylio	3	male	24.5251
12	0.00 : 1.00 → 1	0.64 : 0.36 → 0	0.75 : 0.25 → 0	Jones, Mr. Charl...	1	male	46.00
13	0.67 : 0.33 → 0	0.64 : 0.36 → 0	0.75 : 0.25 → 0	Snyder, Mrs. Jo...	1	female	23.00
14	1.00 : 0.00 → 0	0.68 : 0.32 → 0	0.05 : 0.95 → 1	Howard, Mr. Be...	2	male	63.00
15	0.00 : 1.00 → 1	0.64 : 0.36 → 0	0.05 : 0.95 → 1	Chaffee, Mrs. H...	1	female	47.00
16	0.00 : 1.00 → 1	0.15 : 0.85 → 1	0.05 : 0.95 → 1	del Carlo, Mrs. ...	2	female	24.00
17	1.00 : 0.00 → 0	0.68 : 0.32 → 0	0.05 : 0.95 → 1	Keane, Mr. Daniel	2	male	35.00
18	0.67 : 0.33 → 0	0.49 : 0.51 → 1	0.82 : 0.18 → 0	Assaf, Mr. Gerios	3	male	21.00
19	0.67 : 0.33 → 0	0.49 : 0.51 → 1	0.82 : 0.18 → 0	Ilmakangas, Mis...	3	female	27.00
20	1.00 : 0.00 → 0	0.49 : 0.51 → 1	0.82 : 0.18 → 0	Assaf Khalil, Mr...	3	female	45.00

Hình 4.3 Kết quả dự đoán

Hình sau thể hiện kết quả **dự đoán xác suất sống sót** của 20 hành khách đầu tiên trong tập kiểm tra, thông qua ba mô hình: **kNN**, **Naive Bayes**, và **Decision Tree**. Mỗi mô hình cung cấp:

- Xác suất dự đoán cho mỗi lớp (0 = không sống sót, 1 = sống sót).
- Kết quả cuối cùng (mũi tên → 0 hoặc → 1).
- Các thuộc tính đầu vào: Tên, Pclass, Giới tính (Sex), Tuổi (Age).

Giải thích ý nghĩa của các cột:

- **kNN**: 0.67: 0.33 → 0 → Xác suất sống là 33%, dự đoán không sống sót.
- **Naive Bayes**: 0.49: 0.51 → 1 → Xác suất sống cao hơn, dự đoán sống sót.
- **Tree**: 0.82: 0.18 → 0 → Dự đoán không sống sót với xác suất cao.

Nhận xét từ dữ liệu chi tiết:

Sự khác biệt giữa mô hình:

Có một số hành khách được các mô hình dự đoán **khác nhau**, ví dụ:

- **Wilkes, Mrs. James**: kNN và Naive Bayes dự đoán sống sót (→ `1`), trong khi Tree dự đoán không sống sót (→ `0`).
- **Assaf, Mr. Gerios**: tất cả mô hình dự đoán không sống sót (→ `0`), dù giới tính là nam và tuổi còn trẻ (21).
- **Snyder, Mrs. Joseph**: Tree và kNN đều dự đoán sống (→ `1`), khớp với trực giác vì hành khách là phụ nữ, độ tuổi trẻ.

Vai trò của giới tính (Sex):

Nữ giới có xu hướng được dự đoán sống sót với xác suất cao hơn.

- Ví dụ: "Wilkes, Mrs. James", "Connolly, Miss. Kate", "Abrahim, Mrs. Joseph" – đều được ít nhất 2 mô hình dự đoán là sống.
- Nam giới (đặc biệt thuộc tầng lớp 3 - Pclass 3) như "Kelly, Mr. James", "Davies, Mr. John" thường bị dự đoán là không sống sót.

Vai trò của độ tuổi (Age):

Trẻ em hoặc thanh niên như "Svensson, Mr. Johan" (14 tuổi), "Davies, Mr. John" (21 tuổi),... thường vẫn bị đánh giá thấp khả năng sống nếu là nam và thuộc Pclass 3.

Các hành khách trung niên nữ như "Chaffee, Mrs. Herbert" (47 tuổi) vẫn được dự đoán là sống với xác suất cao.

Mức độ tự tin của mô hình:

Tree thường đưa ra dự đoán chắc chắn hơn (chênh lệch xác suất rõ ràng).

Naive Bayes có xu hướng nghiêng về phân phối sát 50-50 như 0.49 : 0.51, nên dễ bị sai khi ranh giới mờ.

Kết luận từ bảng dự đoán chi tiết:

Qua quá trình thực nghiệm trên tập dữ liệu Titanic bằng phần mềm Orange, sử dụng ba mô hình học máy khác nhau là Decision Tree, k-Nearest Neighbors (kNN) và Naive Bayes, ta rút ra được một số kết luận quan trọng như sau:

Decision Tree là mô hình thể hiện hiệu quả dự đoán tốt nhất. Với độ chính xác cao ở cả hai lớp, đặc biệt là Precision (1) đạt 0.801 – mô hình rất hiệu quả trong việc hạn chế dự đoán sai người không sống sót thành sống sót (FP thấp). Đây là mô hình phù hợp nếu ưu tiên an toàn và giảm thiểu sai lệch nghiêm trọng.

kNN cho thấy sự cân bằng tương đối giữa Precision và Recall. Tuy không vượt trội về chỉ số nào, nhưng lại có ưu điểm ở sự ổn định và dễ điều chỉnh. Mô hình này phù hợp nếu bài toán không nghiêng về một tiêu chí cụ thể nào và cần sự linh hoạt.

Naive Bayes hoạt động kém hiệu quả hơn rõ rệt, đặc biệt ở những trường hợp ranh giới xác suất không rõ ràng. Mặc dù có thể được dùng như baseline (điểm khởi đầu để so sánh), nhưng không được khuyến nghị áp dụng thực tế cho bài toán này.

Ngoài ra, khi quan sát chi tiết từng hành khách, ta thấy rằng giới tính và tầng lớp (Pclass) có ảnh hưởng lớn đến xác suất dự đoán. Điều này cho thấy các mô hình học máy đã học được mối quan hệ quan trọng giữa đặc trưng đầu vào và khả năng sống sót.

Nên ta có thể suy ra rằng:

Decision Tree thường mạnh về xác suất rõ ràng và tỏ ra quyết đoán trong phân loại.

kNN phản ánh sát các đặc trưng gần nhất, nhưng đôi khi đưa ra xác suất trung bình gây do dự.

Naive Bayes thường không vượt trội trong các trường hợp ranh giới xác suất sát nhau.

CHƯƠNG 5. KẾT LUẬN

5.1. Kết quả đạt được

Trong phạm vi đồ án này, nhóm đã hoàn thành toàn bộ quy trình từ xử lý dữ liệu thô đến đánh giá, so sánh và phân tích kết quả dự đoán của các mô hình học máy trên nền tảng trực quan Orange. Cụ thể:

- **Hoàn thiện quy trình xử lý dữ liệu**:

  Tập dữ liệu Titanic gốc chứa nhiều thuộc tính không liên quan (tên, cabin, vé,...), dữ liệu thiếu (tuổi, điểm lên tàu), dữ liệu dạng chuỗi.

  Nhóm đã sử dụng các kỹ thuật tiền xử lý phù hợp: loại bỏ cột dư thừa, mã hóa nhãn (label encoding), xử lý giá trị thiếu bằng trung bình, mode hoặc loại bỏ dòng không cần thiết.

  Tạo ra một tập dữ liệu sạch, cân đối, sẵn sàng đưa vào pipeline học máy.

- **Xây dựng mô hình dự đoán bằng Orange**:

  Tận dụng khả năng kéo-thả trực quan của Orange để thiết lập luồng xử lý dữ liệu rõ ràng.

  Triển khai 3 mô hình học máy cổ điển: **Naive Bayes**, **k-Nearest Neighbors (kNN)** và **Decision Tree**.

  Tách dữ liệu huấn luyện – kiểm tra hợp lý bằng công cụ **Test & Score**, đánh giá kết quả bằng **Confusion Matrix** và bảng thống kê.

- **Đánh giá và phân tích mô hình**:

Tính toán chi tiết các chỉ số **Precision**, **Recall**, **F1-score** cho từng lớp (sống sót/không sống sót).

Thực hiện **so sánh trực tiếp các mô hình**:

- **Decision Tree** đạt Precision cao nhất (0.801), F1-score tốt, phù hợp khi ưu tiên giảm thiểu sai sót nguy hiểm.
- **kNN** thể hiện độ ổn định, cân bằng giữa Precision và Recall.
- **Naive Bayes** hoạt động kém hiệu quả hơn do giả định xác suất chưa phù hợp với phân bố dữ liệu thực tế.

Phân tích kết quả dự đoán trên từng hành khách, từ đó làm rõ tác động của các đặc trưng như giới tính, độ tuổi, tầng lớp lên xác suất sống sót.

Rút ra mô hình phù hợp nhất:

Sau khi đánh giá toàn diện, nhóm xác định rằng **Decision Tree là mô hình phù hợp nhất với bài toán Titanic**, vừa đảm bảo độ chính xác cao, vừa trực quan và dễ lý giải cho người dùng cuối.

Qua đó, nhóm không chỉ thực hiện được mục tiêu ban đầu mà còn nâng cao được kiến thức thực tiễn về **ứng dụng học máy trong bài toán phân loại**, đặc biệt là việc triển khai trên nền tảng trực quan như Orange.

5.2. Những khó khăn, hạn chế

Trong quá trình thực hiện, nhóm cũng gặp phải một số khó khăn và hạn chế như sau:

- **Khó khăn khi xử lý dữ liệu thực tế**:

Dữ liệu ban đầu không sạch, chứa nhiều giá trị thiếu và không đồng nhất.

Việc xử lý chuỗi, mã hóa nhãn và xác định giá trị phù hợp để điền vào chỗ trống đòi hỏi sự cẩn trọng, ảnh hưởng đến chất lượng huấn luyện nếu làm sai.

- **Hạn chế từ công cụ Orange**:

Dù Orange rất trực quan, nhưng **hạn chế trong việc tùy chỉnh sâu** như chọn số lượng lá trong Decision Tree, số lân cận trong kNN, hoặc điều chỉnh các siêu tham số.

Không hỗ trợ dễ dàng các mô hình hiện đại như **Random Forest**, **XGBoost**, hoặc các kỹ thuật nâng cao như **Grid Search**, **Cross-validation nâng cao**,...

- **Phân tích xác suất đầu ra thủ công**:

  Kết quả đầu ra theo từng hành khách rất hữu ích, nhưng việc phân tích phải được thực hiện bằng tay, khó áp dụng cho tập dữ liệu lớn hơn hoặc nếu cần tự động hóa.

- **Chưa triển khai mô hình nâng cao**:

  Do giới hạn về thời gian và phạm vi môn học, nhóm chưa thể triển khai các kỹ thuật học máy tiên tiến hoặc so sánh với mạng nơ-ron, boosting, stacking,...

TÀI LIỆU THAM KHẢO

[1] Kaggle. (n.d.). *Titanic - Machine Learning from Disaster*. https://www.kaggle.com/competitions/titanic

[2] Orange Data Mining. (n.d.). *Official Documentation*. https://orangedatamining.com

[3] Towards Data Science. (n.d.). *Precision, Recall, and F1-Score Explained.* https://towardsdatascience.com/precision-recall-and-f1-score-979ff85ef658

[4] KNN (K-Nearest Neighbors)- https://viblo.asia/p/knn-k-nearest-neighbors-1-djeZ14ejKWz

[5] Bài toán phân lớp dữ liệu trên Organge -môn data science (khoa học dữ liệu)- https://www.youtube.com/watch?v=hHcqmhPs6QE

[6] Thuật toán phân lớp Naive Bayes- https://viblo.asia/p/thuat-toan-phan-lop-naive-bayes-924lJWPm5PM

[7]Decisiontreealgorithmhttps://machinelearningcoban.com/tabml_book/ch_model/decision_tree.html

PHỤ LỤC

Bảng phân công công việc của các thành viên trong nhóm

STT	Nội dung công việc	22DH114826 Lê Phạm Hoàng Vũ	22DH113985 Uông Thành Trung	22DH113984 Trương Đình Trung
1	Chạy thực nghiệm kNN	40%	40%	20%
2	Chạy thực nghiệm Naive Bayes	40%	40%	20%
3	Chạy thực nghiệm DecisionTree	40%	40%	20%
4	Nội dung báo cáo	40%	40%	20%

(Các thành viên trong nhóm cùng tự đánh giá và thống nhất mức độ đóng góp cuả mỗi thành viên cho từng công việc cụ thể)