Journal of Information Systems Education

Volume 36
Issue 4
Fall 2025

Teaching Tip

AI and Machine Learning for Business and Information Systems Education Using KNIME

Mohanad Halaweh

Recommended Citation: Halaweh, M. (2025). Teaching Tip: AI and Machine Learning for Business and Information Systems Education Using KNIME. *Journal of Information Systems Education*, 36(4), 352-366. https://doi.org/10.62273/QOIR5976

Article Link: https://jise.org/Volume36/n4/JISE2025v36n4pp352-366.html

Received: March 21, 2025
First Decision: May 19, 2025
Accepted: July 7, 2025
Published: December 15, 2025

ISSN: 2574-3872 (Online) 1055-3096 (Print)

Journal of Information Systems Education, 36(4), 352-366, Fall 2025
https://doi.org/10.62273/QOIR5976

Teaching Tip
AI and Machine Learning for Business and Information Systems Education Using KNIME

Mohanad Halaweh
College of Business
Al Ain University
Al Ain, United Arab Emirates
mohanad.halaweh@aau.ac.ae

ABSTRACT

Artificial intelligence (AI) and its subfield, machine learning, have become indispensable across various industries. With the aid of low-code/no-code development platform like KNIME, understanding and applying machine learning algorithms has been simplified for various fields, including business and information systems, as these platforms reduce the complexity of necessary technical and coding knowledge. This teaching tip provides a detailed, step-by-step tutorial on applying the machine learning process using KNIME to analyze a healthcare dataset to predict which patients are at risk of diabetes by using classification methods, particularly decision trees. This teaching tip offers a practical, comprehensive, and ready-to-use resource for introducing and understanding machine learning concepts through a low-code platform (KNIME). It also provides valuable insights for practitioners and educators who seek to integrate machine learning into business and information systems curricula.

Keywords: Information systems (IS), IS curriculum, Artificial intelligence, Machine learning, Business data analytics

1. INTRODUCTION

In recent years, artificial intelligence (AI) and machine learning have become essential across various fields, emerging as hot topics in industry and business practice (Abdel-Karim et al., 2021; Perifanis & Kitsios, 2023). These technologies have significant potential to assist in analyzing large volumes of data, provide insights, and facilitate effective decision-making.

However, the information systems (IS) curriculum, as proposed by the Accreditation Board for Engineering and Technology (ABET, 2025), the international accrediting body, lacks specialized courses in AI and machine learning. Typically, programming requirements are limited to a single course, which may be insufficient for preparing students to write code for machine learning algorithms, such as those implemented in Python. Additionally, IS researchers have been slow to adopt machine learning in their studies, contributing to its limited presence in IS curricula. Abdel-Karim et al. (2021) found that, despite the growing popularity of AI and machine learning in industry, these technologies remain underutilized in IS research. Their review of 1,838 articles and a survey of 110 IS researchers highlighted barriers to the adoption of machine learning methods in leading IS journals, notably a lack of expertise in and understanding of machine learning methods within the IS field. This gap may stem from the complexity of the required coding.

While the number of institutions that offer machine learning courses continues to rise, instructional resources that support business and IS educators in teaching these courses remain scarce, particularly regarding the provision of clear guidelines for determining the suitability of machine learning for specific problems. Kayhan (2022) addressed this gap by highlighting the challenge of conceptualizing problems with valid input-output relationships.

Traditionally, machine learning relies on programming languages, such as Python or R, for data processing, querying, and analysis. While effective for software engineering and data science, this approach can be less efficient for business analytics tasks that prioritize generating insights. Visual workflows, such as those provided by KNIME, offer a clearer view of processes, inputs, and outputs without the complexity of coding, aligning more closely with the IS curriculum and the skill development expected of students. This allows business and IS students to focus on transforming data into actionable insights rather than spending excessive time on coding—an endeavor often more suited to computer scientists and software engineers. Platforms like KNIME make machine learning more accessible to business and IS students, as well as practitioners, by minimizing the need for extensive technical knowledge or coding, thereby streamlining the application of powerful machine learning methods to real-world problems.

Recent research indicates a shift among practitioners toward visual programming and low-code/no-code development platforms (LDPs) (Bock & Frank, 2021; Käss et al., 2023; Li et al., 2022; Sundberg & Holmström, 2024; Woo, 2020). These tools cater to both experienced developers and individuals without prior programming experience. Importantly, "low" refers not to the quality of the final application but to the minimal effort required for development, including tools that facilitate workflow automation (Li et al., 2022). KNIME exemplifies a low/no-code data analytics

platform that offers a wide range of machine learning methods and data analytics capabilities.

Currently, there is a notable gap in the literature concerning KNIME and its applications in the IS field. Research in the AIS Electronic Library (AISeL) and Google Scholar focusing on articles published in IS journals revealed a dearth of studies on this topic. Integrating KNIME into courses on business analytics, machine learning, or AI can help students grasp the machine learning process and algorithms in a simplified manner, while also aiding instructors in teaching machine learning concepts through simple practical applications. Although other tutorials and research papers have demonstrated various tools for data analysis and machine learning (e.g., Carvalho et al., 2019; Sundberg & Holmström, 2024), KNIME's visual workflow and intuitive graphical user interface provide distinct advantages, particularly for teaching business and IS students. Additionally, previous efforts did not offer a complete, step-by-step resource that is easily replicated by either independent students or instructors.

Given the limited instructional materials for teaching machine learning in IS, the sparse inclusion of machine learning courses in IS curricula, the industry's shift toward LDPs, the demand for user-friendly visual tools, and the lack of research on using KNIME in the IS literature, this teaching tip paper is both timely and necessary. It presents a detailed, step-by-step tutorial on using KNIME to analyze healthcare datasets, specifically to predict the risk of diabetes. This teaching tip demonstrates how machine learning methods, such as classification using the Decision Tree algorithm, can be effectively applied in the healthcare sector without writing a single line of code. This will prove particularly valuable for instructors teaching machine learning to business and IS majors, as well as for practitioners in data analytics. Finally, this document illustrates how such tools enable users to focus on practical applications and skill development without being overwhelmed by technical complexities.

This teaching tip is structured into seven main sections. Section 2 provides an overview of the machine learning process. Section 3 offers a detailed description of the KNIME platform workspace, familiarizing readers with the tool and its essential functionalities for implementing machine learning. Section 4 presents information about the AI for Healthcare Applications course that used the KNIME platform, along with a step-by-step illustration of the machine learning process applied to Diabetes Risk Prediction using KNIME. Section 5 provides evidence, as reflected in students' feedback and performance, supporting the benefits of using the KNIME tool to achieve the intended course learning outcomes and address the above-mentioned needs. Finally, Section 6 discusses key takeaways and lessons learned, offering guidance to the IS community, followed by the conclusion in Section 7.

2. MACHINE LEARNING PROCESS

Machine learning is a subfield of AI that enables computers or machines to learn from large datasets without being explicitly programmed, which distinguishes the subfield from traditional rule-based programming. This capability allows machines to adapt and enhance their performance over time without manual intervention. The more data fed into a machine learning model, the more it learns and improves.

Machine learning is primarily categorized into supervised learning and unsupervised learning (Abdel-Karim et al., 2021). The former involves training models on labeled data to predict or classify correct labels, commonly by utilizing algorithms such as Decision Trees, Hierarchical Naïve Bayes, and Support Vector Machines. In contrast, unsupervised learning focuses on identifying patterns in unlabeled data, enabling knowledge discovery through techniques like clustering, which might employ, among others, K-Means and Hierarchical Clustering algorithms.

This teaching tip outlines the general step-by-step machine learning process (de Souza Nascimento et al., 2019; Kühl et al., 2021; Sen et al., 2022). The typical steps are as follows:

1) *Problem Definition:* The problem should be clearly defined, as this clarifies the objective of the machine learning model (e.g., to classify, to predict, or to cluster data).
2) *Data Collection:* Relevant data are gathered to address the defined problem. The quality and quantity (size) of the collected data will significantly influence the model's performance.
3) *Data Preprocessing:* The data are cleaned and prepared for use in a machine learning model. This process includes handling missing values, removing duplicates, normalizing or scaling features, and splitting the data into training and test sets (typically, an 80/20 or 70/30 split). Feature selection is also performed to identify the most relevant features, enhancing the model's performance.
4) *Model Selection:* Based on the problem, an appropriate machine learning algorithm is selected; common choices include Decision Trees for classification, Linear Regression for prediction (numerical value), and K-Means for clustering.
5) *Model Training:* The selected model is trained using the training dataset. During this step, the algorithm learns patterns in the data and adjusts its parameters to minimize error.
6) *Model Evaluation:* The model's performance is evaluated using the test dataset (30%) that was not utilized during training. This assessment ensures that the model generalizes well to unseen data. Common metrics for evaluation include accuracy, precision, recall, and specificity score for classification tasks, as well as mean absolute error (MAE), mean squared error (MSE), and R^2 for regression tasks.
7) *Model Deployment:* Once the model exhibits consistently adequate performance, it is deployed in a production environment, where it is used to make predictions on new data. A machine learning model can be utilized directly or integrated with other systems.

3. AN OVERVIEW OF THE KNIME PLATFORM

KNIME (KNIME, 2024a, 2024b) is a comprehensive analytics platform that supports data experts throughout the analytics lifecycle, from data access to deployment and monitoring. It serves as a tool for data analysis, manipulation, visualization, and reporting by offering a consistent low-code/no-code framework within an open-source environment. Users can leverage KNIME for any or all stages of building and deploying an analytics solution. A typical end-to-end process for creating

a data app with KNIME includes the following (KNIME, 2024a, 2024b):

- *Data Access and Integration:* Use KNIME's visual programming environment to access and blend data. It can connect to various technologies in KNIME's open ecosystem, including data warehouses, scripting languages (e.g., R and Python), and machine learning libraries.
- *Model Building:* Construct analytical models in an intuitive visual environment. KNIME allows users to build workflows (see Figure 2) by simply dragging and dropping nodes or components.
- *App Development and Sharing:* Create a data app with an easy-to-use interface, specify permissions, and securely share it via a direct link or an embeddable connection.

This streamlined approach enables users to efficiently develop and deploy analytics solutions without extensive coding.

Figure 1 illustrates a visual workflow node status indicator in KNIME, specifically that of a Partitioning node.

- Inputs: Data enter the node from the left.
- Outputs: Processed data exit the node on the right.
- Status: The node's status is displayed below it using a color-coded system:
  - *Not Configured:* Red indicator, meaning the node setup is incomplete.
  - *Configured:* Yellow indicator; the node is properly set up but not yet executed.
  - *Executed:* Green indicator; the node has successfully run.
  - *Error:* Red with an "X," indicating an execution error.

Each node in KNIME represents a step or task in a data workflow (machine learning/data mining process). It serves as a modular, visual block that performs a specific function, such as data preprocessing, partitioning, filtering, transformation, analysis, model building, or visualization. For instance, the Partitioning node in KNIME (Figure 1) is used to split a dataset into two subsets, allowing for the division of data for tasks such as training and testing machine learning models. This node can, for example, separate 70% of the data for training and 30% for testing.

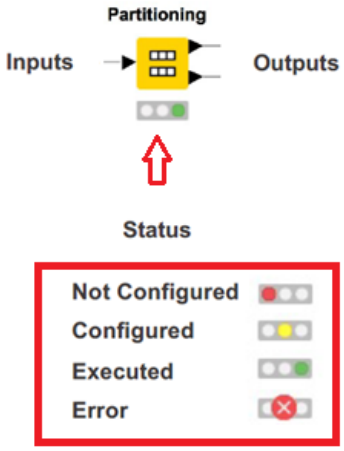


Figure 1. Node in KNIME Workflow

Nodes can be connected to create complex workflows, with each node performing an operation and passing the results to the next node in sequence (Figure 2). This modular design makes it easy to build, understand, and modify data analysis processes without extensive coding. Figure 2 shows the KNIME platform workspace, including different parts of the user interface. The following are brief descriptions of each item highlighted in red in Figure 2:

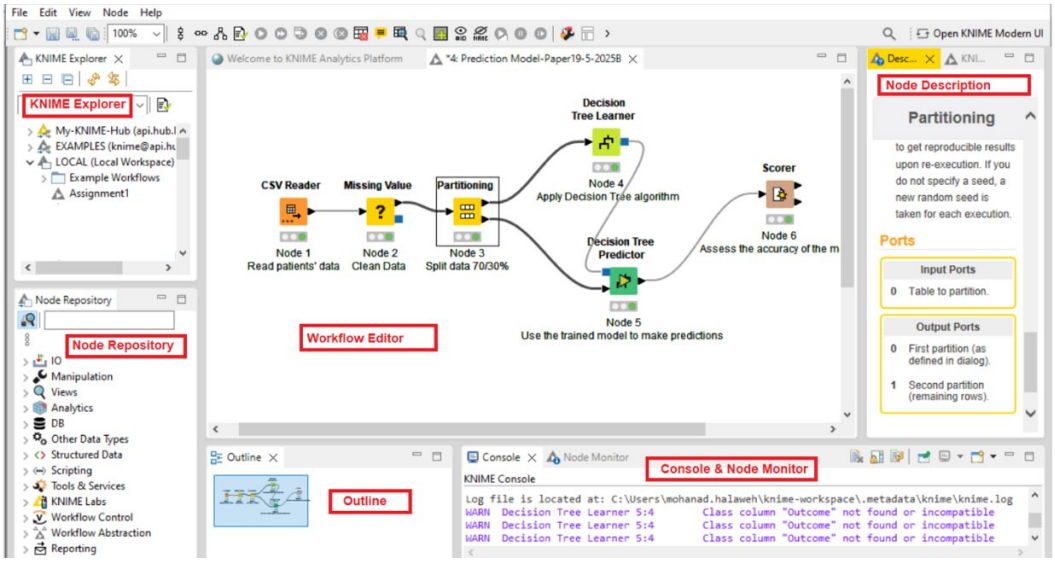


Figure 2. The KNIME Analytics Platform Workbench

- *KNIME Explorer:* This panel displays the structure of the KNIME workspace, showing projects and saved workflows. It facilitates navigation between different projects and workflow management.
- *Node Repository:* This panel contains a library of all available nodes in KNIME, organized by functionality. It allows users to search for and drag nodes into the Workflow Editor to build a workflow. This repository includes all machine learning algorithms, predefined functions, and methods used throughout the analytics and machine learning process, facilitating tasks such as data visualization and preprocessing, as well as model training, evaluation, and deployment.
- *Workflow Editor:* The main area for constructing data workflows. Nodes are dragged and connected to form a visual representation of the analytical/machine learning process.
- *Outline:* A small overview window that provides a zoomed-out view of the entire workflow, making it easier to navigate large or complex workflows.
- *Console & Node Monitor:* Displays errors and warnings related to workflow execution. It provides feedback on node processing and assists in debugging any issues that arise during workflow execution.
- *Node Description:* The panel provides a brief explanation of the selected node's functionality, including the required input and output data, and helps clarify the node's purpose and configuration options.

The appearance of the KNIME software interface may vary by the version downloaded. In the latest edition, users can switch between two interface styles, the default "KNIME Modern UI" and the "Classic User Interface." Figures 6 to 21 use the Classic interface to demonstrate the machine learning steps in this teaching tip. Figure 3 shows the option (under the menu tab) to switch from the Modern UI to the Classic, while Figure 4 shows the option (located in the upper right corner) to switch from the Classic to the Modern UI.

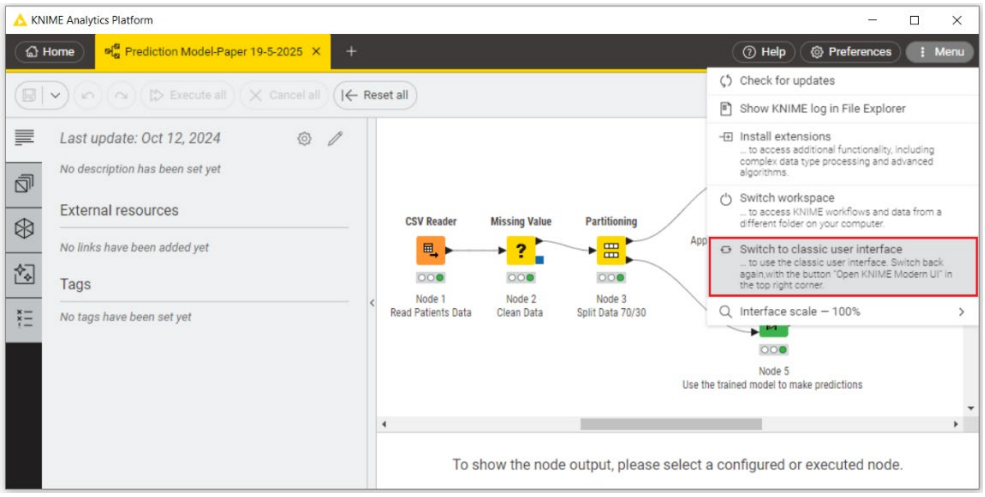


Figure 3. Switching From the KNIME Modern UI to the Classic User Interface

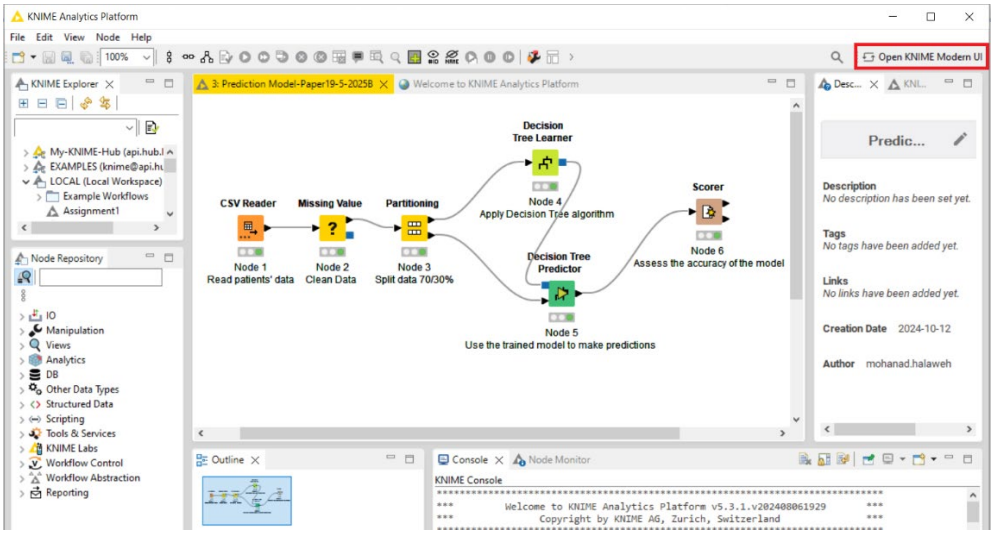


Figure 4. Switching From the KNIME Classic User Interface to the Modern UI

4. COURSE ON AI FOR HEALTHCARE APPLICATIONS WITH KNIME

The teaching tip provided in this paper focuses on using KNIME for diabetes risk prediction. It was implemented in an AI for Healthcare Applications course in a diverse MBA program (with various concentrations). The course covers both theory and application, and the practical component concentrates on the application of machine learning algorithms in a healthcare context. The KNIME platform was demonstrated in multiple tutorial sessions over several weeks; these sessions covered various machine learning and data mining tasks, such as classification, regression, clustering, and association.

A set of learning theories and frameworks, including Bloom's taxonomy, experiential learning, and project-based learning, formed the theoretical foundation for the course and teaching tips presented in this paper. Bloom's (1956) taxonomy facilitated the development of the course tutorials and assessment tools, which focus on higher forms of learning in education, such as applying, analyzing, and evaluating concepts and processes, as well as creating, rather than just remembering, concepts. The KNIME platform was adopted to provide hands-on, practical applications for machine learning and to address, through project assessment, the following course learning outcomes:

- Analyze and evaluate datasets and computational procedures to answer clinical questions.
- Apply AI technologies to develop AI applications in healthcare.
- Formulate a written report and oral presentation to effectively communicate AI-based, data-driven, personalized healthcare solutions for patients.

Course instruction was based on experiential learning, a pedagogical approach designed to facilitate "learning by doing" (Jonathan & Laik, 2024). In the IS field, many studies have shown that experiential learning can lead to a higher level of comprehension, help translate knowledge into skills, and promote lifelong learning outcomes (Triche et al., 2024). Students were provided with tutorials on using the KNIME tool to apply machine learning in a healthcare context (specifically, to predict diabetes cases using a decision tree). Students initially engaged in tutorial sessions and activities by exploring the tool and learning the machine learning process and algorithms through hands-on application and active participation; eventually, they applied models and analyzed data.

In line with project-based learning, which is another form of "learning by doing" that provides additional structure, the course focused on addressing real-world problems and answering questions over an extended period of time, as well as prioritized student autonomy, collaboration, and communication (Kokotsaki et al., 2016). As part of a comprehensive project, students were tasked with acquiring and applying knowledge and skills, using real data (from Kaggle), to address an individually assigned problem in healthcare. They also interpreted the results, reflected on the experience, and provided insights and recommendations. This approach provided them with experiential experience in solving problems and applying machine learning to varied healthcare contexts and challenges (see project description on predicting obesity and the marking rubric in the Appendix).

In this teaching tip, the AI-driven predictive model evaluates the risk of diabetes using machine learning methods, specifically classification via a Decision Tree algorithm. This method is commonly applied in classification tasks, such as predicting the risk of diabetes (target output: diabetes—yes or no), based on various health indicators (age, BMI, blood pressure, etc.). Its primary goal is to facilitate timely diagnosis and enhance treatment planning, ultimately improving overall patient care.

The following steps present the machine learning process applied in the case using KNIME:

- *Problem Definition:* The main objective of the machine learning model is to predict patients who are at risk of diabetes based on health indicators.
- *Data Collection:* Relevant data, including patient age, BMI, and glucose levels, are collected (fictional dataset generated by generative AI for tutorial/educational illustrative purposes).
- *Data Preprocessing:* Cleaning and preparing the data for analysis, as the dataset includes missing values.
- *Model Selection:* Classification method using the Decision Tree algorithm.
- *Model Evaluation:* Assessing the model using a confusion matrix and metrics such as accuracy.
- *Model Deployment:* Integrating the model into a system for real-time or batch processing (not applicable to this teaching tip, as the model can be used to evaluate new patient instances directly in KNIME without integration with a hospital's medical system).

The following are supplementary materials and steps needed to carry out the tutorial. They provide readers with direct access to the resources required to replicate each step.

- **Download KNIME software**: Access the software at https://www.knime.com/downloads
- **Download the CSV file** ("Diabetes Patient Data.csv"): This file is used in KNIME for training and testing the machine learning model. https://drive.google.com/file/d/1F8KGOI1-0YbCALDetrwZteK8nCbACA3y/view?usp=sharing
- **Download the KNIME workflow file** ("Prediction Model.knwf"): This file includes the developed prediction model (resulting model) and can be imported into KNIME. https://drive.google.com/file/d/1a2ciZXwZtHHSd_WQdz9NKT3NN7lJ-dZa/view?usp=sharing
- **Import the workflow** (downloaded resulting model) into the workspace by clicking on the "Import KNIME Workflow..." option, which can be found under the File menu (using the Classic User Interface). If utilizing the Modern UI, the file can be imported by clicking on the Explorer icon and selecting "import workflow" from the dropdown menu, as shown in Figure 5.

Downloading and importing the workflow might be skipped if the workflow is built from scratch, as will be demonstrated next.

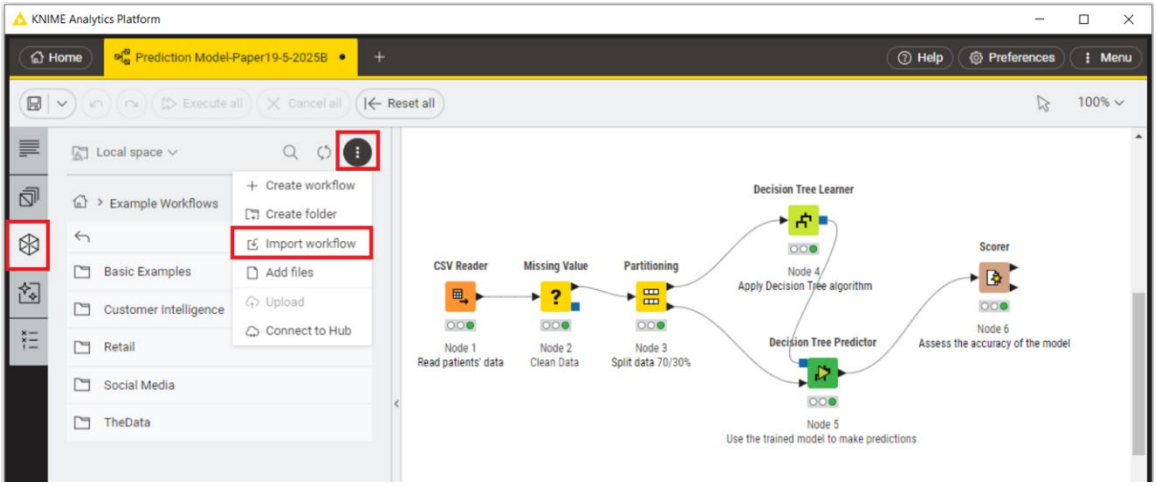


Figure 5. Importing a Created Workflow in KNIME Using the Modern UI

By following the steps demonstrated in this tutorial, even educators and students with no background in machine learning can master the process using KNIME. However, basic knowledge of machine learning and data mining concepts (such as data preparation, cleaning, filtering, data splitting/partitioning, and modeling using decision trees) provides an advantage. In any case, the implementation of these concepts is made easier through KNIME, which does not require a technical understanding of programming complexities.

4.1 KNIME Workflow for Diabetes Risk Prediction

Figure 6 shows the workflow for Diabetes Risk Prediction created using KNIME. The following is a description of each node in the workflow and its function within the machine learning process.

4.1.1 CSV Reader (Node 1). Function: This step reads data from a CSV file (see supplementary materials), which serves as the starting point for any machine learning task. In this context, it imports a dataset related to patients, including features such as age, blood pressure, glucose levels, and other relevant indicators for predicting diabetes. To configure the CSV Reader node, users must navigate to and select the downloaded Diabetes Patient Data CSV file. Figure 7 shows the configuration dialog of the CSV Reader (Node 1) in the KNIME Analytics Platform, which is used to import the CSV file into the workflow. The output from this node, after execution, is a table containing 780 patient records, as illustrated in Figure 8.

4.1.2 Missing Value (Node 2). Function: This step addresses any missing or incomplete data in the dataset. It ensures that the dataset is clean and ready for analysis by filling in or removing missing values, which is crucial for accurate machine learning model training. As shown in Figure 9, the data table imported in the previous step contains missing values, as indicated in KNIME by a red question mark ("?").

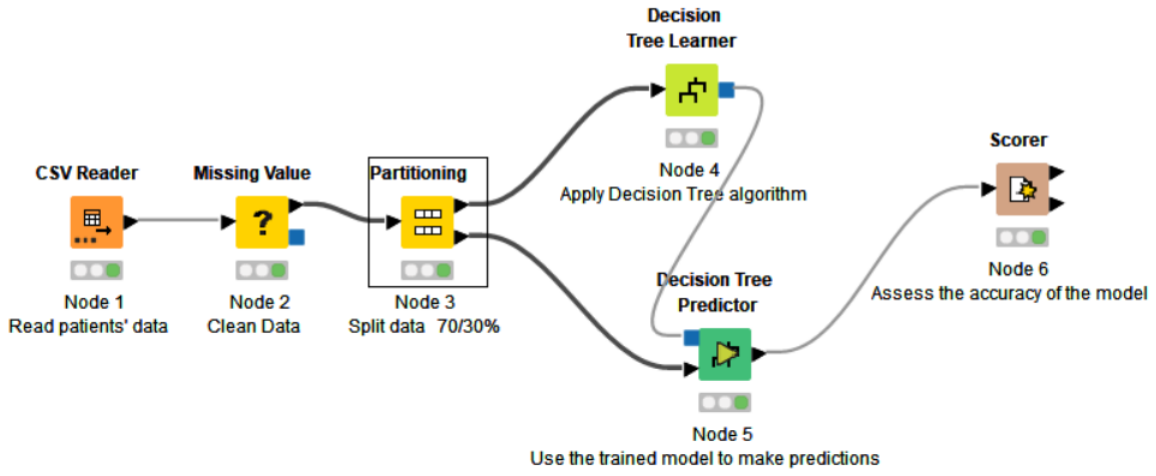


Figure 6. KNIME Workflow for Diabetes Risk Prediction

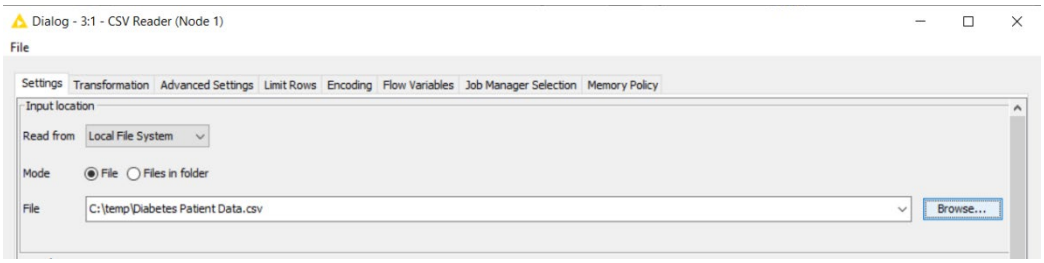
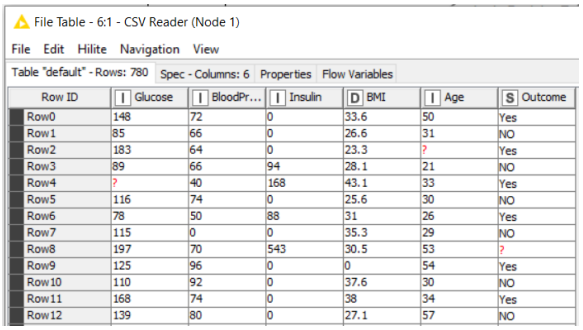
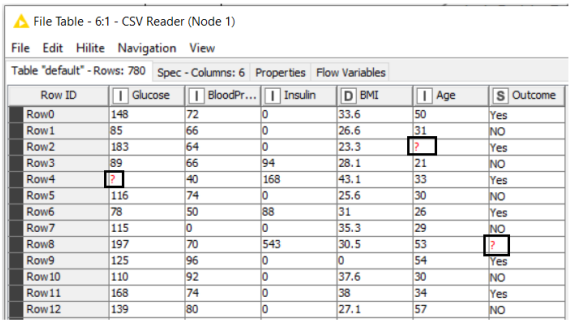


Figure 7. CSV Reader Node Configuration in KNIME



Row ID	Glucose	BloodPr...	Insulin	BMI	Age	Outcome
Row0	148	72	0	33.6	50	Yes
Row1	85	66	0	26.6	31	NO
Row2	183	64	0	23.3	?	Yes
Row3	89	66	94	28.1	21	NO
Row4	?	40	168	43.1	33	Yes
Row5	116	74	0	25.6	30	NO
Row6	78	50	88	31	26	Yes
Row7	115	0	0	35.3	29	NO
Row8	197	70	543	30.5	53	?
Row9	125	96	0	0	54	Yes
Row10	110	92	0	37.6	30	NO
Row11	168	74	0	38	34	Yes
Row12	139	80	0	27.1	57	NO

Figure 8. Sample Dataset (780 Patient Records) for Diabetes Risk Prediction Output From CSV Reader Node



Row ID	Glucose	BloodPr...	Insulin	BMI	Age	Outcome
Row0	148	72	0	33.6	50	Yes
Row1	85	66	0	26.6	31	NO
Row2	183	64	0	23.3	?	Yes
Row3	89	66	94	28.1	21	NO
Row4	?	40	168	43.1	33	Yes
Row5	116	74	0	25.6	30	NO
Row6	78	50	88	31	26	Yes
Row7	115	0	0	35.3	29	NO
Row8	197	70	543	30.5	53	?
Row9	125	96	0	0	54	Yes
Row10	110	92	0	37.6	30	NO
Row11	168	74	0	38	34	Yes
Row12	139	80	0	27.1	57	NO

Figure 9. Dataset (Sample) With Missing Values for Diabetes Prediction Analysis in KNIME

Figure 10 displays the settings for managing missing values in a dataset within KNIME's Missing Value node (Node 2). Various strategies can be applied, including replacing missing glucose values with the maximum observed value, removing rows with missing outcome data (where "Yes" indicates a diagnosis of diabetes and "No" indicates the absence of diabetes), and filling missing age values with the mean age. For example, if age is missing, an expert might decide to fill in the average age value. However, if critical values, such as the outcome (presence "Yes" or absence of diabetes "No") or insulin levels, are missing, the entire record should be removed, as this could significantly impact the model's accuracy.

Figure 11 displays the outcome from Node 2: a table with cleaned data. As shown, one record was removed due to a missing outcome value, reducing the total number of rows to 779. This adjustment reflects the configuration to eliminate rows with missing outcome values (row 8). Additionally, the age for the patient in row 2 was filled with the mean age of 33.27, while the missing glucose value in row 4 was replaced with the maximum value within the range.

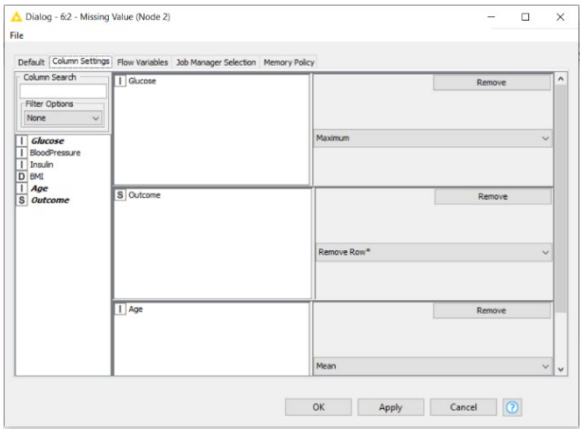
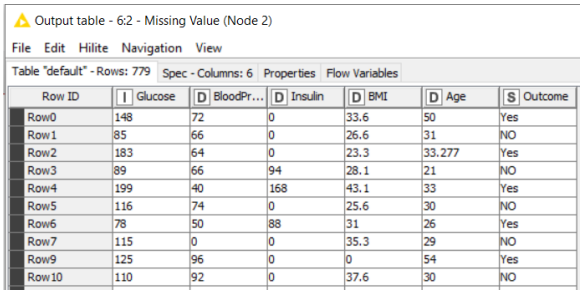


Figure 10. Missing Value Handling Configuration in KNIME



Row ID	Glucose	BloodPr...	Insulin	BMI	Age	Outcome
Row0	148	72	0	33.6	50	Yes
Row1	85	66	0	26.6	31	NO
Row2	183	64	0	23.3	33.277	Yes
Row3	89	66	94	28.1	21	NO
Row4	199	40	168	43.1	33	Yes
Row5	116	74	0	25.6	30	NO
Row6	78	50	88	31	26	Yes
Row7	115	0	0	35.3	29	NO
Row9	125	96	0	0	54	Yes
Row10	110	92	0	37.6	30	NO

Figure 11. Outcome (Sample) After Handling Missing Value in Dataset

4.1.3 Partitioning (Node 3). Function: This step splits the dataset into training and testing sets. This partitioning is essential for training the machine learning model on one portion of the data (the training set) while testing its performance on unseen data (the test set) to evaluate accuracy and reliability.

Figure 12 displays the configuration for partitioning the dataset in KNIME using the Partitioning node. The specific settings are as follows: First, the dataset is split into two partitions, with the first partition set to 70% of the dataset using a relative percentage. Next, the data points are selected using the "Draw randomly" method to ensure a random sampling.

Finally, a random seed value of 1 is used, enabling consistent and reproducible results across multiple runs.

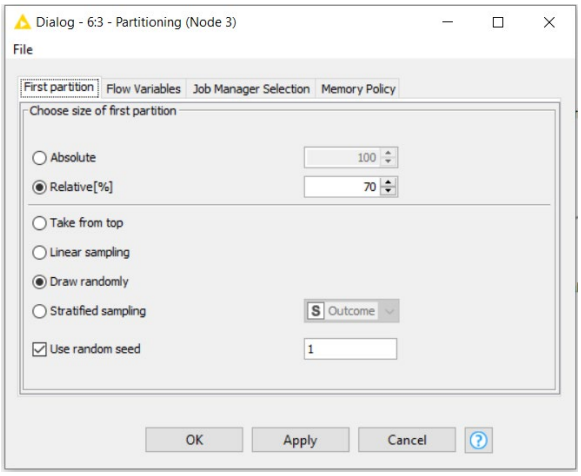


Figure 12. Partitioning Configuration for Data Splitting in KNIME

This setup is designed to provide a randomized yet consistent split of the data for training and testing purposes. Alternative options can be selected; for example, 70% of the data could be chosen by selecting the "Take from top" option for training.

Figure 13 displays the first partition of the dataset (the first output of Node 3), as defined in the partitioning configuration in KNIME. The dataset is split according to the specified settings, resulting in 545 rows (70% of the original dataset), including attributes such as glucose, blood pressure, insulin, BMI, age, and outcome. This partition is typically used for training the machine learning model.

First partition (as defined in dialog) - 6:3 - Partitioning (Node 3)

File Edit Hilite Navigation View

Table "default" - Rows: 545 Spec - Columns: 6 Properties Flow Variables

Row ID	Glucose	BloodPr...	Insulin	BMI	Age	Outcome
Row0	148	72	0	33.6	50	Yes
Row1	85	66	0	26.6	31	NO
Row2	183	64	0	23.3	33.277	Yes
Row3	89	66	94	28.1	21	NO
Row7	115	0	0	35.3	29	NO
Row10	110	92	0	37.6	30	NO
Row11	168	74	0	38	34	Yes
Row12	139	80	0	27.1	57	NO
Row13	189	60	846	30.1	59	Yes
Row17	107	74	0	29.6	31	Yes

Figure 13. First Partition of Dataset (Sample) After Splitting in KNIME

Figure 14 displays the second partition of the dataset (the second output of Node 3), which contains the remaining 234 rows (30% of the original dataset). It includes the same columns: glucose, blood pressure, insulin, BMI, age, and outcome. This partition is typically used as the test set to evaluate the performance of the machine learning model.

Second partition (remaining rows) - 6:3 - Partitioning (Node 3)

File Edit Hilite Navigation View

Table "default" - Rows: 234 Spec - Columns: 6 Properties Flow Variables

Row ID	Glucose	BloodPr...	Insulin	BMI	Age	Outcome
Row4	199	40	168	43.1	33	Yes
Row5	116	74	0	25.6	30	NO
Row6	78	50	88	31	26	Yes
Row9	125	96	0	0	54	Yes
Row14	166	72	175	25.8	51	Yes
Row15	100	0	0	30	32	Yes
Row16	118	84	230	45.8	31	Yes
Row21	99	84	0	35.4	50	NO
Row31	158	76	245	31.6	28	Yes
Row34	122	78	0	27.6	45	NO
Row39	111	72	207	37.1	56	Yes
Row46	146	56	0	29.7	29	NO
Row50	103	80	82	19.4	22	NO

Figure 14. Second Partition of Dataset (Sample) After Splitting in KNIME

4.1.4 Decision Tree Learner (Node 4). Function: This trains a decision tree model using the training dataset. The Decision Tree algorithm learns patterns in the data to make predictions, identifying the features that most significantly influence the outcome, such as predicting whether a patient has diabetes.

Figure 15 illustrates the configuration used in the Decision Tree Learner node in KNIME. The class column, which serves as the target for prediction in supervised learning, is set to "Outcome," indicating the presence or absence of diabetes. The chosen quality measure for the decision tree is the Gini index, which is used to determine optimal splits. The root split can be enforced on a specific attribute, such as the glucose column, although other attributes or indicators can also be selected as the root for splitting, depending on the analysis requirements and domain knowledge.

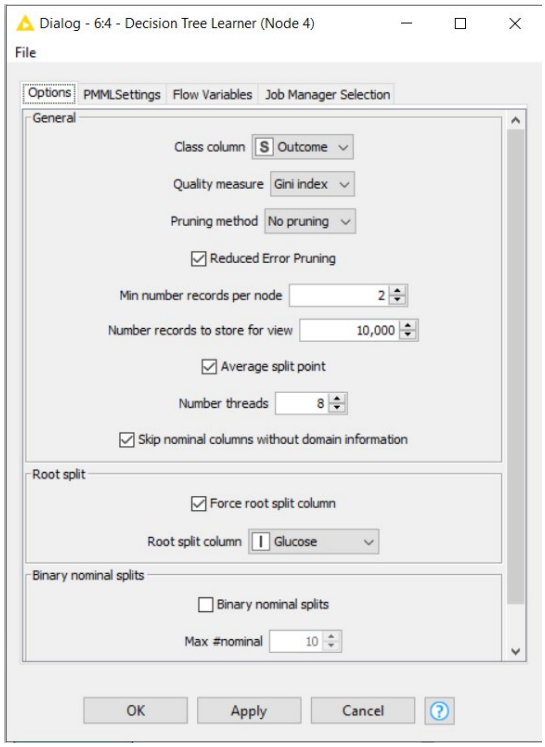


Figure 15. Decision Tree Configuration in KNIME Using Decision Tree Learner Node

Figure 16 displays a visualization of the decision tree generated using the Decision Tree Learner node in KNIME. This tree structure is based on various health indicators, such as glucose, BMI, and insulin, to predict the presence or absence of diabetes (Outcome). Key elements of the decision tree are as follows:

- The root node is determined by the glucose value (147.5), highlighting its importance in diabetes prediction.
- Subsequent splits are made using attributes like BMI, age, and insulin levels.
- Each node contains a table that shows the category distribution (Yes/No for diabetes), along with respective counts and percentages, providing insight into the classification decisions at each split.

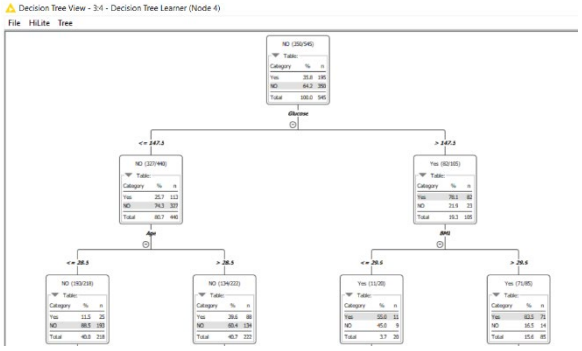
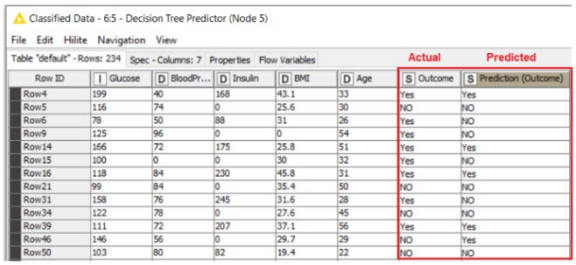


Figure 16. Decision Tree (Partial) Visualization for Diabetes Prediction in KNIME

While Figure 16 shows a zoomed-in view of a portion of the decision tree, Figure 17 presents the entire decision tree for a comprehensive overview.

4.1.5 Decision Tree Predictor (Node 5). Function: This node uses the trained decision tree model to make predictions on the test set, evaluating the model's ability to generalize by predicting outcomes based on data that were not used during training. As illustrated in the workflow in Figure 6, Node 5 has two inputs, the trained model and the test data used for predictions. The output of the node is the input table (test dataset) with one addition: a new column named Prediction (Outcome) that contains the predicted values, resulting in the output shown in Figure 18.



Row ID	Glucose	BloodPr...	Insulin	BMI	Age	Outcome (Actual)	Prediction (Outcome) (Predicted)
Row4	199	40	168	43.1	33	Yes	Yes
Row5	116	74	0	25.6	30	NO	NO
Row6	78	50	88	31	26	Yes	NO
Row9	125	96	0	0	54	Yes	NO
Row14	166	72	175	25.8	51	Yes	Yes
Row15	100	0	0	30	32	Yes	NO
Row16	118	84	230	45.8	31	Yes	Yes
Row21	99	84	0	35.4	50	NO	NO
Row31	158	76	245	31.6	28	Yes	NO
Row34	122	78	0	27.6	45	NO	NO
Row39	111	72	207	37.1	56	Yes	Yes
Row46	146	56	0	29.7	29	NO	Yes
Row50	103	80	82	19.4	22	NO	NO

Figure 18. Classified Data (Sample Test Dataset) Using the Decision Tree Predictor in KNIME

4.1.6 Scorer (Node 6). Function: This node assesses the accuracy of the predictions made by the model. It compares the predicted values to the actual outcomes (shown in Figure 18), providing a confusion matrix and metrics related to accuracy, precision, and recall to evaluate the model's performance.

Figure 19 shows the configuration settings of the Scorer node in KNIME, which is used to evaluate the performance of a predictive model. The actual outcomes are set to the "Outcome" column (the first column), while the predicted outcomes are specified in the "Prediction (Outcome)" column (the second column). This setup allows for a comparison between the actual class and the predicted class.

Figure 20 displays the confusion matrix output from the Scorer node (Node 6), summarizing the performance of the predictive model for diabetes classification.

- **True Positives (Yes, Yes)**: 49 cases where the model correctly predicted "Yes" for diabetes.
- **False Positives (No, Yes)**: 29 cases where the model incorrectly predicted "Yes" for diabetes when it was actually "No."
- **False Negatives (Yes, No)**: 30 cases where the model incorrectly predicted "No" for diabetes when it was actually "Yes."
- **True Negatives (No, No)**: 126 cases where the model correctly predicted "No" for diabetes.

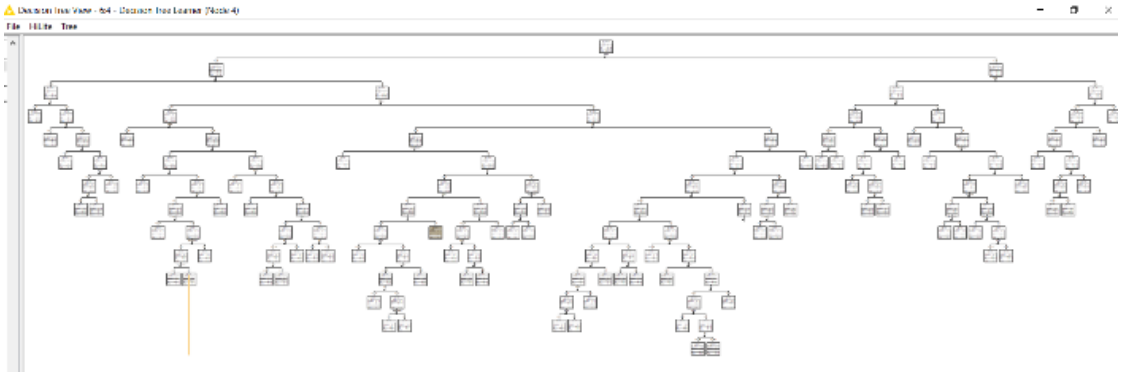


Figure 17. Entire Decision Tree in KNIME

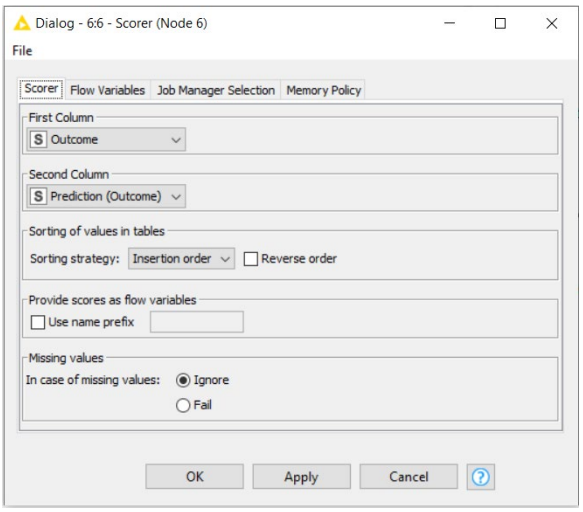


Figure 19. Scorer Node Configuration in KNIME for Model Evaluation

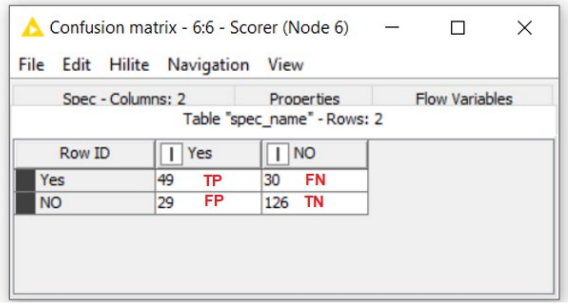


Figure 20. Confusion Matrix for Model Evaluation in KNIME

Calculating the total (49 + 29 + 30 + 126) results in 234 cases, which was the number used to test the model (see Figure 14). The confusion matrix helps evaluate the accuracy, sensitivity, specificity, and overall effectiveness of the predictive model.

Figure 21 provides a detailed summary of the accuracy statistics from the Scorer node, assessing the performance of the predictive model for diabetes classification. The table includes the following metrics:

- **Overall Accuracy (74.8%)**: The model's accuracy was 0.748, indicating that approximately 74.8% of predictions (both positive and negative cases) were correct. However, accuracy alone is not sufficient for evaluating prediction quality and should be complemented with the following metrics (for the Yes class).
- **Precision (62.8%):** This measures the percentage of predicted positive cases that are actually positive. A precision of 62.8% means that, of all cases predicted to be positive, 62.8% of those predictions were correct. The remaining 37.2% were false positives (cases that were predicted to be positive but were actually negative). These results might lead to unnecessary follow-up tests or procedures, which could waste resources and increase patient anxiety due to unnecessary follow-ups.
- **Recall (62%)**: This measures the percentage of actual positive cases that are correctly identified. The model successfully identified 62% of actual diabetes cases, indicating a moderate ability to detect true positives. This means that while some diabetes cases were detected, nearly 38% of true cases were not. Improving recall could enhance early detection, enable timely interventions, reduce complications, and potentially lower healthcare costs.
- **Sensitivity (62%)**: This is equivalent to recall, as it measures the percentage of actual positive cases that are correctly identified. Sensitivity is particularly important in medical research and clinical settings, where accurately identifying positive cases is critical.
- **Specificity (81.3%)**: This measures the percentage of actual negative cases that are correctly identified. The model accurately identified 81.3% of non-diabetic cases as negative, demonstrating a good rate of true negatives. High specificity ensures that non-diabetic patients are not misclassified. This can enhance patient trust in the diagnostic process, alleviate unnecessary stress for those incorrectly flagged, and minimize resource waste.

The accuracy results produced by the KNIME software can also be calculated manually using the confusion matrix results (see Figure 20). This manual calculation can be included in a tutorial for students to clarify how the results are derived. Table 1 presents the calculations based on the relevant metric formulas.

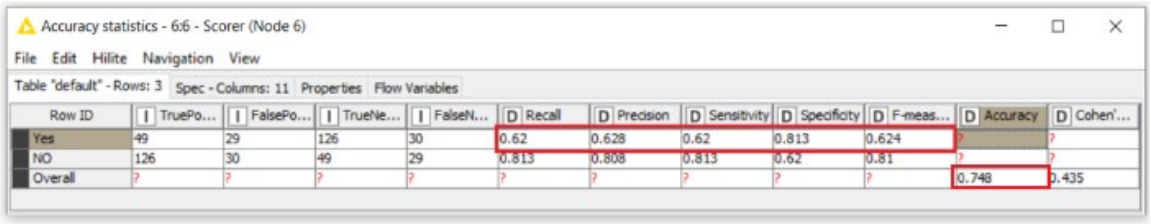


Figure 21. Accuracy Statistics for Model Evaluation in KNIME

Metric	Formula	Calculation	Value
Accuracy	(TP + TN) / (TP + FP + FN + TN)	(49 + 126) / (49 + 29 + 30 + 126)	0.748
Precision	TP / (TP + FP)	49 / (49 + 29)	0.628
Recall (Sensitivity)	TP / (TP + FN)	49 / (49 + 30)	0.62
Specificity	TN / (TN + FP)	126 / (126 + 29)	0.813

Table 1. Model Evaluation Metrics (Yes Class)

Given that the model evaluation results were not highly satisfactory, the analyst might consider several options to enhance performance. These could include increasing the size of the analyzed dataset, modifying the splitting attribute used in the decision tree, or improving data quality by sourcing more reliable information. Alternatively, the analyst could explore switching to a different classification algorithm altogether (e.g., Random Forest) to achieve better results.

After delivering multiple sessions demonstrating the machine learning process, including the case above and other applications of various algorithms using KNIME, students were assigned a project (using data from Kaggle) in which they applied the knowledge they had gained through the tutorial sessions.

5. EVIDENCE

The tutorial presented in the teaching tip was implemented in an AI for Healthcare Applications course in an MBA program (with different concentrations) that included students from various backgrounds. After completing the tutorial sessions, which spanned several weeks and covered various machine learning and data mining algorithms, the students were assigned a project, which 11 students completed.

The project assignment focused on a different healthcare problem to assess their knowledge and the skills they had acquired. Working on the project provided them with hands-on experience in applying machine learning algorithms and techniques using the KNIME tool. The details of the project and the rubric are attached in the Appendix.

Many of the students who took the course shared positive feedback directly with the instructor, both during the semester and via email after completion: "*It was a great experience to take the AI application in healthcare course with you. I learned so much from the course, and I liked using the KNIME software. If possible, I would like to ask for your advice as I would like to know more about AI applications in healthcare, especially with the KNIME software. How can I improve myself and learn more about this? How can I convince my future workplace to start applying AI? Thank you so much for your effort in this course. You made it easy for me to understand—and I believe my other classmates feel the same—since I come from a medical background and I am not well-educated in AI or even MIS, but your teaching made things simple and easy to understand, so I just wanted to thank you, Professor, for that.*"

Regarding objective evidence that students benefited from this course, the teaching approach, and the use of the KNIME platform during the practical portion of the course, as shown in Table 2, the students performed well in the project assessment (with details of the project and rubric attached in the Appendix), achieving an average score of 18.36 out of 20. This indicates that they met the expected learning outcomes.

Student ID	Project Score (of 20)
Student1	18
Student2	17
Student3	19
Student4	18
Student5	19
Student6	18
Student7	19
Student8	19
Student9	19
Student10	19
Student11	17

Table 2. Student Performance in the Project Assignment

In addition to excellent performance in the assessment, the students also evaluated the instructor's teaching performance and the course using a survey questionnaire on a 5-point Likert scale. Only the section of the survey related to the course is included here. This survey was conducted neutrally by the Quality Assurance & Institutional Research Center and was provided to the instructor after the course was completed. Table 3 summarizes the results of that survey.

As shown, the students reported learning new skills and concepts and found the course content useful and appropriate. Furthermore, they provided written feedback reflecting their experiential learning in the course. Students showed interest in the course content and expressed enthusiasm about further exploring AI and its applications. The feedback (through email) also suggests that KNIME facilitated the application of machine learning methods and algorithms without the complexity of coding. The students indicated (through email and written feedback) a desire to learn more about other AI courses and the KNIME tool.

6. TEACHING SUGGESTIONS AND DISCUSSION

The integration of machine learning into business and IS curricula is essential and can be simplified by using tools that emphasize application over programming. The focus should be on applying machine learning in business contexts and selecting appropriate algorithms rather than on developing or coding new algorithms. If the curriculum requires students to master coding, especially through writing Python code for machine learning algorithms, it risks blurring the lines between IS and other fields, such as AI, software engineering, and data science. While programming is not unnecessary, if a platform like KNIME can accomplish a task without coding, it is often more efficient to use it. Coding should be reserved for complex tasks that the platform cannot handle. Fortunately, KNIME is capable of managing complex tasks for both educational and real-world applications.

Question (5-Point Likert Scale)	Percent of Agreement/ Satisfaction
I learned new ideas and skills from the course	100
The instructional materials offered in the course were appropriate	100
The course was based on a variety of learning resources	90
Assessment tools were appropriate, clear, and covered the whole course content	98
Assignments were useful to understand the course content	94
The course content was clear, useful, and contributed to understanding the academic program	98
Generally speaking, the course can be considered distinguished compared to other courses	96.65
Average	*96.66*
Students' Feedback/Comments	
"Learnt a lot from the AI ML model. Was an eye opener and a good experience"	
"Wonderful teaching method with good examples"	
"AI is the new trend worldwide and it was a very great opportunity to learn about AI and its applications in the healthcare sector"	
"Make AI one of the fundamental courses in the program because it is the current trend now and everyone should know how to use it and benefit from it in the different sectors"	
"I wanted to know more about AI and I wish that there are more subjects related to AI"	

Table 3. Student Evaluation of the Course

Even if programming in Python for machine learning is required, instructors can begin by teaching students the process using the KNIME platform. Once students visually engage with inputs, outputs, and functions, they will find it easier to transition to programming, and once they grasp concepts like data cleaning, data splitting, classification models (e.g., decision trees), model evaluation, and confusion matrices, they can apply these when coding in Python for the course.

A visual workflow is often more effective than hundreds of lines of Python code when it comes to reading, cleaning, standardizing data, developing models, testing, evaluating, and visualizing results. Simplicity is powerful; complexity is not always essential to solving difficult problems. With KNIME, results can be obtained instantly, and configuration adjustments can be made quickly and easily without coding.

IS practitioners should prioritize using LDPs like KNIME while emphasizing an understanding of business problems, objectives, requirements, data quality, and data governance—factors that significantly impact the accuracy of machine learning outcomes. They should focus on decision-making, insights gained from data analytics, and implications for the business domain, such as the impact of accurate results on healthcare services and patient care, as highlighted in the previous section. Thus, there should be less emphasis on coding, as development can be effectively managed through LDPs.

The KNIME tool provides a wide range of supervised and unsupervised algorithms. While this tutorial demonstrated a classification example using a decision tree, other classification algorithms and machine learning methods, such as regression, association, and clustering, can be applied using the same process with different algorithm nodes. The tutorial steps can be adapted by other IS instructors and researchers to implement different algorithms with variations based on the algorithm's parameters and inputs. The node descriptions (see Figure 2) and instant documentation that details the node, its inputs, outputs, and functions make it easy to apply a machine learning algorithm.

This step-by-step tutorial is ready for use by instructors and students. These materials can be seamlessly integrated into AI, machine learning, or data/business analytics courses, providing a practical resource for understanding key concepts. The structured approach ensures that learners can easily follow along, enhancing their comprehension of the machine learning process and data analysis without the need for advanced coding skills. This accessibility aligns well with the educational goals in business and IS programs. Instructors may need to explain certain concepts, such as the confusion matrix and accuracy, including how they are calculated manually, to align with the results obtained from KNIME. This additional explanation will help students better understand the outcomes.

7. CONCLUSION

This teaching tip not only provides a step-by-step guide to applying a machine learning model but also highlights the rationale for using an LDP. A workflow that can be created in just a few minutes with KNIME might otherwise require hours of coding and debugging, making it challenging to grasp concepts that are easily visualized through a graphical user interface, as illustrated in this paper.

The simplicity, speed, and ease of use offered by the KNIME software are greatly beneficial for understanding machine learning. By significantly reducing the time and workload for students, the tool enhances learning efficiency and facilitates faster comprehension. This tutorial provides instructors with a structured, hands-on approach to machine learning that is accessible to business, IS, and other students (such as data science or computer science students), especially in business and IS programs where exposure to programming is minimal. While IS instructors can utilize this tutorial in relevant courses, it also offers opportunities for other IS researchers and educators to explore the capabilities of KNIME for more advanced machine learning and deep learning algorithms, such as neural networks.

8. ACKNOWLEDGMENTS

The author would like to thank the Editor-in-Chief, Senior Editor, and the reviewers for their valuable feedback on this paper.

9. REFERENCES

Abdel-Karim, B. M., Pfeuffer, N., & Hinz, O. (2021). Machine Learning in Information Systems—A Bibliographic Review and Open Research Issues. *Electronic Markets,* 31(3), 643-670. https://doi.org/10.1007/s12525-021-00459-2

ABET. (2025). *Criteria for Accrediting Computing Programs, 2022-2023.* https://www.abet.org/accreditation/accreditation-criteria/criteria-for-accrediting-computing-programs-2025-2026/

Bloom, B. S. (1956). *Taxonomy of Educational Objectives, Handbook I: The Cognitive Domain.* New York: David McKay.

Bock, A. C., & Frank, U. (2021). Low-Code Platform. *Business & Information Systems Engineering*, 63, 733-740. https://doi.org/10.1007/s12599-021-00726-8

Carvalho, A., Levitt, A., Levitt, S., Khaddam, E., & Benamati, J. (2019). Off-the-Shelf Artificial Intelligence Technologies for Sentiment and Emotion Analysis: A Tutorial on Using IBM Natural Language Processing. *Communications of the Association for Information Systems*, 44. https://doi.org/10.17705/1CAIS.04443

de Souza Nascimento, E., Palheta, M. P., Ahmed, I., Steinmacher, I., Oliveira, E., & Conte, T. (2019). Understanding Development Process of Machine Learning Systems: Challenges and Solutions. *2019 IEEE/ACM 1st International Workshop on Machine Learning Systems Engineering* (pp. 39-45). https://doi.org/10.1109/ESEM.2019.8870157

Jonathan, L. Y., & Laik, M. N. (2024). Using Experiential Learning Theory to Improve Teaching and Learning in Higher Education. *European Journal of Education*, 7(2), 18-33. https://doi.org/10.26417/ejser.v6i1.p123-132

Kayhan, V. (2022). When to Use Machine Learning: A Course Assignment. *Communications of the Association for Information Systems,* 50. https://doi.org/10.17705/1CAIS.05005

Käss, S., Strahringer, S., & Westner, M. (2023). Practitioners' Perceptions on the Adoption of Low Code Development Platforms. *IEEE Access*, 11, 29009-29034. https://doi.org/10.1109/ACCESS.2023.3258539

Kokotsaki, D., Menzies, V., & Wiggins, A. (2016). Project-Based Learning: A Review of the Literature. *Improving Schools*, 19(3), 267-277. https://doi.org/10.1177/1365480216659733

Kühl, N., Hirt, R., Baier, L., Schmitz, B., & Satzger, G. (2021). How to Conduct Rigorous Supervised Machine Learning in Information Systems Research: The Supervised Machine Learning Report Card. *Communications of the Association for Information Systems*, 48. https://doi.org/10.17705/1CAIS.04845

Li, M. M., Peters, C., Poser, M., Eilers, K., & Elshan, E. (2022). ICT-Enabled Job Crafting: How Business Unit Developers Use Low-Code Development Platforms to Craft Jobs. *Proceedings of the 43rd International Conference on Information Systems*, 16. https://aisel.aisnet.org/icis2022/is_futureofwork/is_futureofwork/16

KNIME. (2024a). A Step-by-Step Guide to Building Data Apps. https://www.knime.com/files/a-step-by-step-guide-to-building-data-apps-with-knime.pdf

KNIME. (2024b). KNIME Data Apps Beginners Guide. https://docs.knime.com/latest/data_apps_beginners_guide/#introduction

Perifanis, N. A., & Kitsios, F. (2023). Investigating the Influence of Artificial Intelligence on Business Value in the Digital Era of Strategy: A Literature Review. *Information*, 14(2), 85. https://doi.org/10.3390/info14020085

Triche, J., Dong, T., Landon, J., & Baied, E. (2024). Teaching Tip: Enhancing Student's Understanding of Enterprise Systems Using Salesforce. *Journal of Information Systems Education*, 35(1), 25-36. https://doi.org/10.62273/LSVK6170

Sen, R., Heim, G., & Zhu, Q. (2022). Artificial Intelligence and Machine Learning in Cybersecurity: Applications, Challenges, and Opportunities for MIS Academics. *Communications of the Association for Information Systems*, 51. https://doi.org/10.17705/1CAIS.05109

Sundberg, L., & Holmström, J. (2024). Teaching Tip: Using No-Code AI to Teach Machine Learning in Higher Education. *Journal of Information Systems Education*, 35(1), 56-66. https://doi.org/10.62273/CYPL2902

Woo, M. (2020). The Rise of No/Low-Code Software Development—No Experience Needed? *Engineering*, 6(9), 960-961. https://doi.org/10.1016/j.eng.2020.07.007

AUTHOR BIOGRAPHY

Mohanad Halaweh is a professor of information systems (IS) at Al Ain University, United Arab Emirates (UAE). He is an active researcher with over 60 publications in internationally ranked journals and conferences, all of which are indexed in SCOPUS and/or ABDC. He has participated in many top-tier international conferences in the IS field, such as ICIS, PACIS, and ACIS. Additionally, he has received several research awards. Dr. Halaweh has served as a program committee member and reviewer for various international journals and conferences, as well as a speaker and chair at international conferences. His research interests focus on e-commerce and digital innovation, IT adoption, and the impacts of emerging technologies such as AI and big data.

APPENDIX

Project Assignment

Project: Applying Machine Learning Process to Predict Obesity Using KNIME Software

Project Context: Obesity is a significant global health issue that affects millions of people. It can lead to serious health complications like heart disease, diabetes, and other chronic conditions. Healthcare providers are increasingly using data-driven methods and tools like machine learning to predict obesity and intervene early, helping patients manage their weight and avoid future health problems.

In this project, you are required to use the KNIME software to develop a machine learning model to classify whether patients are obese or not, based on their health and lifestyle indicators. Using KNIME, you will build a classification model (such as a Decision Tree) to predict obesity from the provided dataset of 2,000 patients.

Project Objective: The objective of this project is to give you hands-on experience in applying machine learning algorithm and techniques to healthcare data in, using the KNIME analytics platform.

Dataset Overview: The dataset consists of 2,000 patient records (obtained from Kaggle), each with the following features:

- Gender: Male or Female
- Age: Patient's age in years
- Height: Height in meters
- Weight: Weight in kilograms
- Family History: Whether the patient has a family history of being overweight
- Frequent Consumption of High-Calorie Food (FAVC)
- Vegetable Consumption (FCVC)
- Number of Main Meals (NCP)
- Physical Activity Frequency (FAF)
- Time Spent Using Technology (TUE)
- Consumption of Alcohol (CALC)
- and other indicators
- Obesity Classification (ObesityLevel): This is the target variable you will be predicting, classifying patients as insufficient weight, normal weight, overweight level I, overweight level II, obesity type I, obesity type II, obesity type III.

Deliverables:

A. Report on the process of applying machine learning to develop a prediction model for obesity
This report should include:

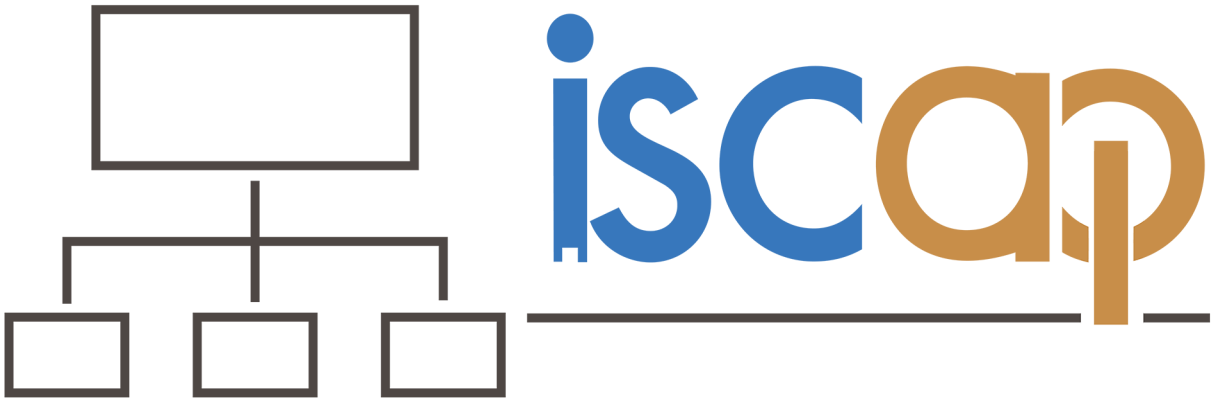
1. Introduction to the Problem: A discussion of the issue of obesity and the purpose of developing the machine learning model.
2. Workflow (Model Development): A complete KNIME workflow that illustrates data preparation (including handling missing data), building a classification model (Decision Tree), model training, and evaluation. Additionally, include the generated decision tree in your report. Visualize the data using basic tools such as histograms and charts to understand the data and the relationships between features.
3. Model Evaluation Results: An evaluation and explanation of how well the model performed, including accuracy statistics and the confusion matrix. Discuss possible improvements and provide examples and results from KNIME.
4. Insights: Key insights from the analysis, including a discussion on the importance of specific features in predicting obesity. Provide recommendations for healthcare professionals and other stakeholders, along with conclusions and reflections.

B. Presentation: A brief presentation summarizing your workflow, results, and recommendations for healthcare professionals.

Marking Criteria and Rubric:

Criteria	2 Marks (Excellent)	1 Mark (Satisfactory)	0 Marks (Unsatisfactory)	Max Mark	Student Mark	Remarks
Implementation of Machine Learning Process (Predictive Model) and Application Through KNIME Software **30%**	Successfully develops a predictive model, clearly demonstrating the modeling process, including data preparation, and selection, training, and evaluation of the machine learning algorithm.	Develops a predictive model with some understanding but shows a basic understanding of machine learning concepts, lacks depth or clarity in some areas.	Does not develop a predictive model or provides a model that is irrelevant or poorly constructed for predicting obesity. Does not demonstrate an understanding of machine learning concepts.	**6**		
Data Analysis, Evaluation and Interpretation **30%**	Thoroughly analyzes the dataset, effectively uses data preprocessing techniques, and provides insightful interpretations of results related to obesity prediction, and model improvement.	Performs basic data analysis or fail to interpret results clearly.	Fails to analyze the dataset or provide any meaningful interpretations related to obesity prediction.	**6**		
Application of Findings (Insights and Recommendations) **20%**	Propose actionable health interventions or insights based on model predictions, demonstrating the potential impact on obesity prevention and management.	Show findings but lacks depth in proposing interventions or insights, or they are not well-aligned with the model predictions.	Does not provide any actionable insights related to obesity prevention or management.	**4**		
Presentation and Clarity **20%**	Presents the project (report and orally) in a clear, organized, and engaging manner, with appropriate use of visuals, well-structured sections, and coherent explanations.	Presentation (report and orally) is somewhat clear but lacks organization or may have unclear explanations; visuals may not effectively support the content.	Presentation (report and orally) is poorly organized, confusing, or lacks coherence, making it difficult to understand the project.	**4**		
Total				**20**		

INFORMATION SYSTEMS & COMPUTING ACADEMIC PROFESSIONALS



STATEMENT OF PEER REVIEW INTEGRITY

All papers published in the ***Journal of Information Systems Education*** have undergone rigorous peer review. This includes an initial editor screening and double-blind refereeing by three or more expert referees.

ISSN: 2574-3872 (Online) 1055-3096 (Print)