IJEEC

International Journal of Electrical Engineering and Computing
Vol. 10, No. 1 (2026)

Original research paper
UDC
DOI

Enhancing MLOps for Educational Data Mining: A Comparative Study of Weka and KNIME in Model Lifecycle Management

Ognjen P. Tomić[1], Miloš Ž. Papić[2]

[1] *LOLA INSTITUT d.o.o. Belgrade, Serbia*
[2] *Department of Industrial Management*, Faculty of Technical Sciences Čačak, Serbia

E-mail address: ognjen.tomic@li.rs, milos.papic@ftn.kg.ac.rs

Abstract—This paper presents a comparative case study of Weka and KNIME for supporting Machine Learning Operations principles in Educational Data Mining. The goal of the study is to evaluate how two widely used data mining platforms support both analytical modeling and model lifecycle management in an educational data analysis context. The study uses the Ocisceni_Demografski_Podaci.csv dataset, containing approximately 15,000 demographic and education-related records from 42 municipalities. For the experimental evaluation, a cleaned analytical subset of 256 instances and 12 attributes was prepared and used consistently in both platforms. Both platforms were evaluated using the same preprocessing logic, classification, clustering, visualization, and lifecycle-management tasks. The analytical evaluation included Naive Bayes, k-nearest neighbors, K-Means, and FarthestFirst algorithms, while the lifecycle evaluation assessed workflow reproducibility, pipeline automation, deployment readiness, and integration extensibility using a five-level maturity rubric. Weka classifier performance was evaluated using stratified 10-fold cross-validation, whereas KNIME classifier results were obtained through a hold-out workflow and are interpreted within the corresponding platform-specific evaluation setting. The results show that Weka provides strong support for rapid algorithmic experimentation and model comparison, with the evaluated classifiers achieving classification accuracy between 80.08% and 82.81%, with KNN (k = 10) producing the highest accuracy under the selected evaluation protocol. In contrast, KNIME provides stronger support for reproducible workflow construction, automation, integration with external systems, and operationalization through server- or hub-based infrastructure. The findings indicate that neither platform is universally superior; rather, their usefulness depends on the phase of the machine learning lifecycle. The study concludes that educational institutions can benefit from a hybrid approach in which Weka is used for early-stage model exploration, while KNIME is used for workflow reconstruction, automation, reporting, and reproducible operationalization.

Keywords—Machine Learning Operations, Educational Data Mining, Weka, KNIME, Workflow Reproducibility, Model Lifecycle Management, Data Pipeline Automation, Comparative Case Study

I. INTRODUCTION

Educational institutions are increasingly data-rich environments, generating vast information on student demographics, performance, and engagement. Educational Data Mining (EDM) leverages this data to improve learning outcomes and optimize resource allocation [1]. However, transitioning from experimental models to production-level systems requires Machine Learning Operations (MLOps) practices [5, 6].

MLOps extends DevOps principles to machine learning, encompassing the entire lifecycle from data preparation to deployment and monitoring [5,7]. Recent MLOps literature emphasizes that the purpose of MLOps is to support reliable and efficient deployment, operation, and maintenance of machine learning models in production environments [5, 7, 18, 19]. This research investigates how accessible data analytics platforms facilitate MLOps for EDM, focusing on Waikato Environment for Knowledge Analysis (Weka) and KNIME Analytics Platform (KNIME).

The primary research question is: How do Weka and KNIME compare in their inherent support for key MLOps principles within educational data analysis contexts?

To answer this question, the study compares Weka and KNIME across two complementary dimensions: analytical capability and MLOps lifecycle capability. Analytical capability refers to the ability to preprocess data, train models, evaluate results, and support exploratory EDM tasks. MLOps lifecycle capability refers to the ability to reproduce, automate, deploy, integrate, and monitor model workflows.

The main contributions of this paper are as follows:

This paper is a revised and expanded version of the paper presented at the XXV International Symposium INFOTEH-JAHORINA 2026

- a comparative evaluation framework for assessing Weka and KNIME against MLOps lifecycle requirements in Educational Data Mining; an empirical case study based on approximately 15,000 demographic and education-related records from 42 municipalities;
- a five-level maturity scoring rubric for evaluating workflow reproducibility, pipeline automation, deployment readiness, and integration extensibility; a practical hybrid Weka–KNIME workflow recommendation for educational institutions that need both rapid model experimentation and reproducible operationalization.

Previous tool comparisons focused on algorithmic performance and usability. Al-Radaideh et al. [9] compared classification methods across several data mining tools, while Jović et al. [8] evaluated free software tools for general data mining tasks. However, these studies predate the MLOps paradigm.

Recent MLOps research has produced architectural overviews, lifecycle definitions, taxonomies, maturity models, and tool-support surveys [5, 7, 16, 17, 19, 21]. Continuous development and deployment of AI models further require pipeline-oriented support across data handling, model learning, software development, and system operations [18].

II. RELATED WORK

A. Educational Data Mining

Educational Data Mining (EDM) is an interdisciplinary research field that applies data mining, machine learning, statistics, and learning analytics techniques to data generated in educational environments. Its main objective is to discover patterns that can support decision-making related to student performance, institutional planning, resource allocation, dropout prediction, curriculum improvement, and learning personalization. Romero and Ventura define EDM as a field concerned with developing methods for exploring unique types of educational data and using those methods to better understand students and educational settings [1]. In addition, Baker and Yacef emphasize that EDM has evolved from exploratory analysis of student interaction data toward predictive and prescriptive analytics that can support institutional interventions [2]. More recent EDM and learning analytics surveys further show that the field has expanded toward institutional analytics, data-driven education, educational data science, and broader decision-support scenarios [23]. Within this context, demographic and educational datasets are particularly relevant because they enable institutions and public-sector bodies to analyze enrollment trends, identify municipalities or schools exposed to demographic decline, and plan educational resources more efficiently. However, many EDM studies remain focused on model development and evaluation, while less attention is given to the operational lifecycle of models after their initial creation.

B. Weka and KNIME as Data Mining Platforms

Weka and KNIME are widely used platforms for data mining and machine learning, but they differ significantly in their design philosophy and operational capabilities. Weka is a mature data mining workbench that provides a broad collection of algorithms for classification, regression, clustering, association rule mining, attribute selection, and model evaluation [3]. Its Explorer, Experimenter, and Knowledge Flow interfaces make it suitable for rapid prototyping, algorithm comparison, and educational use, especially in environments where users need immediate access to standard machine learning methods with minimal configuration. KNIME, on the other hand, is based on a visual workflow paradigm in which analytical processes are represented as connected nodes for data access, preprocessing, modeling, visualization, and reporting. Fillbrunn et al. highlight KNIME's ability to support reproducible and cross-domain analytical workflows through modular pipeline construction and integration with external tools [4]. Therefore, while Weka is particularly strong in algorithm-level experimentation, KNIME is better aligned with workflow-level process management, integration, and repeatability.

C. MLOps and Model Lifecycle Management

Machine Learning Operations (MLOps) extends DevOps principles to machine learning systems by addressing the full lifecycle of data preparation, model training, validation, deployment, monitoring, governance, and continuous improvement. Kreuzberger et al. define MLOps as a set of practices that aims to deploy and maintain machine learning models reliably and efficiently in production environments [5]. Unlike traditional software systems, machine learning systems introduce additional operational risks because their behavior depends not only on code, but also on data, features, model parameters, training pipelines, and changing real-world distributions. Sculley et al. describe these issues as hidden technical debt in machine learning systems, emphasizing that experimental ML code often becomes difficult to maintain when deployed without proper lifecycle management [6]. Recent MLOps surveys further show that tool support remains fragmented, with different platforms addressing only specific parts of the lifecycle, such as experiment tracking, pipeline orchestration, model serving, monitoring, or data validation [7]. For this reason, evaluating data mining platforms through an MLOps lens is important, particularly in domains such as education where models must be reproducible, auditable, and maintainable over time.

D. Research Gap

Previous studies have compared data mining platforms such as Weka, KNIME, RapidMiner, Orange, Tanagra, and R primarily in terms of usability, algorithm availability, classification performance, and general analytical functionality [8, 9]. However, most of these comparisons were conducted before the widespread adoption of MLOps and therefore do not evaluate whether these tools can support reproducible, automated, deployable, and monitorable machine learning workflows. Similarly, existing MLOps studies usually focus on cloud-native or code-centric platforms and provide limited discussion of traditional data mining tools used by domain experts in educational institutions. This creates a gap between EDM model development and operational model lifecycle management. The present study addresses this gap by comparing Weka and KNIME not only as data mining platforms, but as potential components of an MLOps-oriented

EDM workflow. The comparison is specifically scoped to four lifecycle dimensions: workflow reproducibility, pipeline automation, deployment readiness, and integration extensibility. In this way, the study contributes a practical evaluation framework for understanding how accessible visual analytics tools can support the transition from experimental EDM models to maintainable operational systems.

III. MATERIALS AND METHODS

A. RESEARCH DESIGN AND APPROACH

This study employed a comparative case study design informed by MLOps lifecycle literature and empirical software engineering guidelines [5, 7, 12]. The research was conducted in three phases:

- tool capability assessment;
- practical implementation;
- MLOps maturity evaluation.

This multi-phase approach allowed for both theoretical and practical validation of findings. The comparison was not designed to determine which platform is universally superior, but to evaluate their suitability for different phases of an MLOps-oriented EDM lifecycle.

B. DATASET AND PREPROCESSING

The study utilized the Ocisceni_Demografski_Podaci.csv dataset, which contains approximately 15,000 demographic and education-related records from 42 municipalities. The dataset includes municipal identifiers, demographic indicators, and educational indicators related to population structure and enrollment-relevant categories. The dataset was selected because it enables both analytical evaluation, through classification and clustering tasks, and lifecycle-management evaluation, through the implementation of equivalent workflows in Weka and KNIME.

For experimental evaluation, the original dataset was cleaned, transformed, and reduced to an analytical subset of 256 instances with 12 selected attributes. Each experimental instance represents an aggregated demographic record defined by municipality, local community, and sex category. The reduced subset was used as the direct modeling input in both WEKA and KNIME, whereas the original dataset of approximately 15,000 records was retained as the source dataset for preparation and validation.

The source attribute used to construct the Kategorija target variable was excluded from the predictor set to prevent target leakage.

Data preprocessing followed the CRISP-DM methodology [20], while data-quality checks were aligned with established data validation practices for machine learning pipelines [10]. The preprocessing included:

- verification of missing and inconsistent records;
- handling missing numerical values using mean imputation;
- encoding categorical variables, including municipality names and gender indicators;
- normalization of numerical attributes using min-max scaling;
- validation of consistency across demographic and educational categories;
- preparation of equivalent input formats for Weka and KNIME.

The same preprocessing logic was applied in both platforms to ensure that the comparison focused on platform capabilities rather than differences in data preparation.

Table 1 summarizes the main characteristics of the dataset used in the case study.

TABLE I. *DATASET* CHARACTERISTICS

Property	Value
Dataset name	Ocisceni_Demografski_Podaci.csv
Number of records	approximately 15,000
Experimental dataset	256 cleaned instances and 12 attributes
Number of municipalities	42
Data type	demographic and education-related records
Main attribute groups	municipal identifiers, demographic indicators, educational indicators
Analytical tasks	classification and clustering
Classification target	education-related category derived from enrollment indicators
Evaluation protocol	WEKA: stratified 10-fold cross-validation; KNIME: hold-out train–test partitioning.
Missing value treatment	mean imputation for numerical features
Categorical encoding	municipality names, gender indicators, and other categorical attributes
Normalization	min-max scaling for numerical attributes

The classification task was defined as predicting an education-related category derived from the available enrollment and demographic indicators. This formulation reflects the educational planning objective of identifying municipalities with different demographic and enrollment profiles. The same class attribute and preprocessing procedure were used in both Weka and KNIME to ensure comparability of analytical results.

C. EXPERIMENTAL SETUP

The experimental framework was designed to evaluate both platforms across identical analytical and lifecycle-management tasks. The same dataset, preprocessing logic, and class definition were used in Weka and KNIME. However, Weka classifiers were evaluated using stratified 10-fold cross-validation, whereas KNIME classifiers were evaluated through a hold-out train–test workflow. Therefore, the reported classifier metrics are interpreted within their respective platform-specific evaluation settings and are not treated as a direct cross-platform accuracy benchmark.

The analytical workflow consisted of the following steps:

- data ingestion and validation;
- feature selection and transformation;

- descriptive statistical analysis;
- model training and validation;
- results visualization and interpretation.

Algorithm implementation included standard classification and clustering methods commonly used in data mining workflows [11]:

- Classification: Naive Bayes and k-nearest neighbors (KNN), with k = 10 and k = 50;
- Clustering: K-Means with k = 3 clusters and the FarthestFirst algorithm;
- Classification metrics: accuracy, precision, recall, and F1-score;
- Clustering outputs: cluster assignments, cluster-size distributions, centroid/profile summaries, and visual inspection.

For exploratory clustering, K-Means and FarthestFirst were applied after min–max normalization. The results were interpreted descriptively through cluster assignments, cluster-size distributions, centroid or profile summaries, and visual inspection. No silhouette coefficient or external cluster-validity metric was calculated. The same class attribute was used in both Weka and KNIME to ensure that differences in results were attributable to platform behavior and workflow implementation rather than to different target definitions. Classification performance was evaluated using accuracy, precision, recall, and F1-score.

The experimental configuration used for both platforms is summarized in Table 2.

TABLE II. *EXPERIMENTAL* CONFIGURATION

Component	Value
Classification algorithms	Naive Bayes; KNN, k = 10; KNN, k = 50
Clustering algorithms	K-Means, k = 3; FarthestFirst
Classification metrics	Accuracy, precision, recall, F1-score
Clustering outputs	cluster assignments, cluster-size distributions, centroid/profile summaries, and visual inspection.
Evaluation protocol	WEKA: stratified 10-fold cross-validation; KNIME: hold-out train–test partitioning.
Random seed	SimpleKMeans seed: 42; FarthestFirst: default platform seed.
Weka version	3.8.7
KNIME Analytics Platform version	5.8.3
Operating system	Windows 11
CPU	Intel Core i7-6700 @ 3.40 GHz
RAM	32 GB
Java version	Java SE 24
Dataset format	CSV input; ARFF export used for WEKA experiments

Weka classifiers were evaluated using stratified 10-fold cross-validation, whereas KNIME classifiers were evaluated using a hold-out train–test partitioning workflow. Therefore, classifier metrics are interpreted within each platform-specific experimental workflow and are not treated as a direct cross-platform accuracy benchmark. Under the WEKA stratified 10-fold cross-validation protocol, the evaluated classifiers achieved accuracy values between 80.08% and 82.81%, as shown in Fig. 5. Since the primary objective of the study was platform comparison rather than predictive-performance optimization, default algorithm configurations were used unless otherwise specified.

D. EVALUATION FRAMEWORK

The MLOps capability assessment was structured around four primary dimensions derived from established MLOps literature, taxonomy studies, maturity-model research, tool-support surveys, and empirical software engineering guidelines [5, 7, 12, 16, 17, 21]:

- **Workflow Reproducibility**: Ability to version, share, and recreate entire analytical pipelines;
- **Pipeline Automation**: Support for automated execution, scheduling, and triggering;
- **Deployment Readiness:** Native capabilities for model packaging, serving, and integration;
- **Integration & Extensibility**: Support for external systems, application programming interfaces (APIs), and custom extensions.

TABLE III. *MLOPS CAPABILITY SCORES*

Dimension	Criterion	Weka	KNIME
Workflow reproducibility	Version control	2	4
Workflow reproducibility	Documentation	3	5
Workflow reproducibility	Environment reproducibility	2	4
Pipeline automation	Scheduling	2	4
Pipeline automation	Error handling	2	4
Pipeline automation	Conditional execution	1	4
Deployment readiness	Model serving	2	4
Deployment readiness	Monitoring	1	3
Deployment readiness	Scalability	2	4
Integration and extensibility	Data sources	3	5
Integration and extensibility	Custom code	4	5
Integration and extensibility	Community ecosystem	4	5

The detailed capability scores are presented in Table 3, while the maturity scoring rubric is shown in Table 4.

TABLE IV. *TABLE 4. MLOPS MATURITY SCORING RUBRIC*

Score	Meaning
1	Feature absent or requires fully manual external implementation
2	Basic feature exists but is incomplete or not integrated into lifecycle workflow
3	Feature is available with manual configuration or external plugin
4	Feature is natively supported but requires enterprise/server setup or additional configuration
5	Feature is natively supported, documented, reproducible and directly usable in MLOps workflow

Disagreements between evaluators were resolved through discussion and documented in an evaluation log. To reduce subjectivity, each maturity score was assigned only when the corresponding capability was implemented or verified in official documentation.

For each platform, the overall MLOps maturity score was calculated as the arithmetic mean of the four dimension-level scores:

$$M = (R + A + D + I) / 4$$

where M denotes the overall MLOps maturity score, R denotes workflow reproducibility, A denotes pipeline automation, D denotes deployment readiness, and I denotes integration and extensibility.

Each dimension was evaluated using a 5-point maturity scale (1=Basic, 5=Advanced) based on observable capabilities and documented features.

E. PLATFORM VERSIONS AND ENVIRONMENT

Both platforms were tested on identical hardware (Intel i7, 32GB RAM, Windows 11) and were configured with default settings. All workflows were implemented twice by different researchers to ensure reproducibility and identify platform-specific learning curves.

IV. RESULTS AND DISCUSSION

A. WEKA: THE ALGORITHMIC SPECIALIST

Weka demonstrated strong capabilities in the model development phase, consistent with previous descriptions of Weka as a mature data mining workbench [3], standard data mining methodology [11], and comparative studies of data mining tools [9]. As shown in Fig. 1, the Weka Explorer environment was used for dataset inspection, preprocessing, and exploratory analysis prior to model development. The interface provides access to attribute statistics, class distributions, and preprocessing functions, allowing researchers to validate data quality and prepare datasets for machine learning experiments.

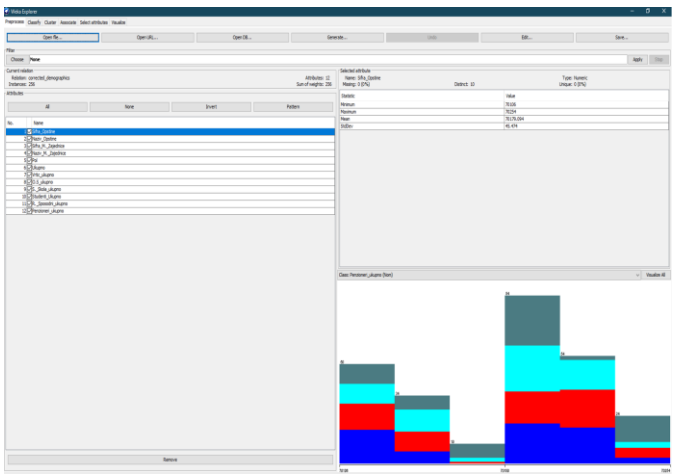

Figure 1. Weka Explorer interface with the processed demographic dataset

As shown in Fig. 1, the Weka Explorer environment was used to load, inspect, and preprocess the demographic and education-related dataset employed in this study. The interface provides an integrated view of dataset attributes, descriptive statistics, class distributions, and preprocessing operations.

The processed dataset contained 256 aggregated instances and 12 attributes representing municipal identifiers, demographic indicators, and education-related variables. Weka enabled rapid inspection of attribute properties, validation of data consistency, and preparation of the dataset for subsequent classification and clustering experiments.

The graphical distribution presented in the lower-right section of Fig. 1 illustrates the class structure of the analyzed dataset, while the statistics panel provides descriptive information for the selected attribute. These capabilities support exploratory data analysis and facilitate the early stages of the machine learning lifecycle, particularly data understanding and preparation.

The platform's Explorer interface provided immediate access to a broad collection of classification and clustering algorithms, together with comprehensive evaluation metrics. Strengths are identified:

- Rapid algorithm prototyping with minimal configuration;
- Extensive built-in algorithm library requiring no additional dependencies;
- Intuitive results visualization for model comparison;
- Strong support for educational demographic classification tasks, with Naive Bayes achieving 80.08% accuracy under the selected 10-fold cross-validation protocol.

The reported WEKA classification result was obtained using the same preprocessed dataset and class definition as in KNIME, but under the stratified 10-fold cross-validation protocol specific to the WEKA experiments. Since the objective of the study was not algorithm optimization, the result is used primarily to demonstrate analytical feasibility and to support the comparison of lifecycle-management capabilities. However, significant MLOps limitations were observed:

- **Workflow Reproducibility**: The Knowledge Flow interface offered basic pipeline construction but lacked native version control integration. Workflow changes required manual documentation;
- **Pipeline Automation**: Limited to basic sequential execution without support for conditional logic or error handling;
- **Deployment Readiness**: Model export produced serialized Java objects (.model files) requiring custom integration code for production deployment;
- **Monitoring**: No built-in capabilities for model performance tracking or data drift detection.

The implementation of the Naive Bayes algorithm in Weka, depicted in Fig. 2, demonstrates the platform's approach to algorithm configuration and result visualization.

These findings align with recent MLOps surveys noting that traditional academic tools often cover model-development phases effectively but lack robust deployment support [4].

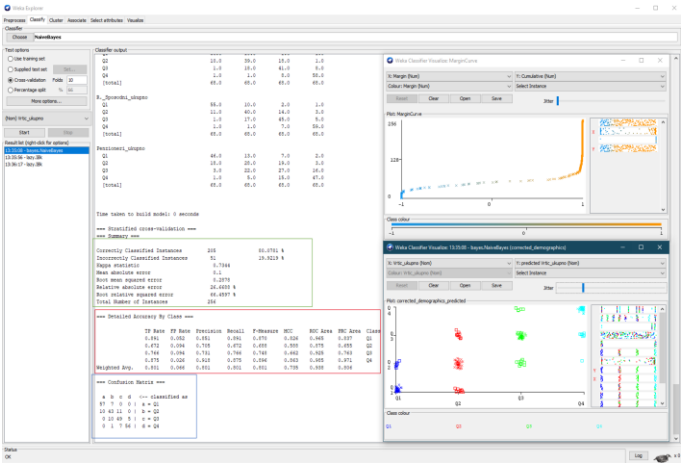

Figure 2. Naive Bayes Classification Results

The implementation of the Naive Bayes classifier in Weka is presented in Fig. 2. The experiment was performed using 10-fold cross-validation on the processed demographic and education-related dataset containing 256 aggregated instances.

The obtained results indicate that Naive Bayes correctly classified 205 instances, achieving an overall classification accuracy of 80.08%. The weighted average precision, recall, and F1-score were approximately 0.80, demonstrating stable predictive performance across the evaluated educational categories.

The confusion matrix shown in Fig. 2 reveals that most classification errors occurred between neighboring demographic categories, while the highest classification performance was achieved for classes Q1 and Q4. The visualization panel further illustrates the relationship between actual and predicted class assignments, providing additional insight into class separability and model behavior.

These results confirm that Naive Bayes can serve as an effective baseline classifier for educational demographic analysis while maintaining low computational complexity and fast model construction times.

The classification results obtained using the K-Nearest Neighbors (KNN) algorithm with k = 10 are presented in Fig. 3. The model was evaluated using a 10-fold cross-validation procedure on the processed demographic dataset containing 256 instances and 12 selected attributes.

The KNN classifier correctly classified 212 instances, achieving an overall classification accuracy of 82.81%, which represents the highest predictive performance among the evaluated machine learning models. The weighted average precision, recall, and F1-score reached approximately 0.83, indicating strong classification capability and balanced performance across all analyzed categories.

The confusion matrix demonstrates that most observations were correctly assigned to their respective classes, with limited misclassification occurring primarily between neighboring demographic groups. The visualization panel shows a clear separation between the four target categories (Q1–Q4), suggesting that the selected demographic and educational features provide sufficient discriminatory information for neighborhood-based classification.

The margin curve presented in the upper-right section of Fig. 3 indicates that the majority of instances were classified with positive confidence margins. Compared with the Naive Bayes model, the KNN classifier achieved improved class separation and a higher overall classification accuracy, demonstrating the effectiveness of distance-based learning for this dataset.

These findings suggest that KNN with k = 10 provides an appropriate balance between local pattern recognition and generalization, making it the most effective classification approach among the evaluated models.

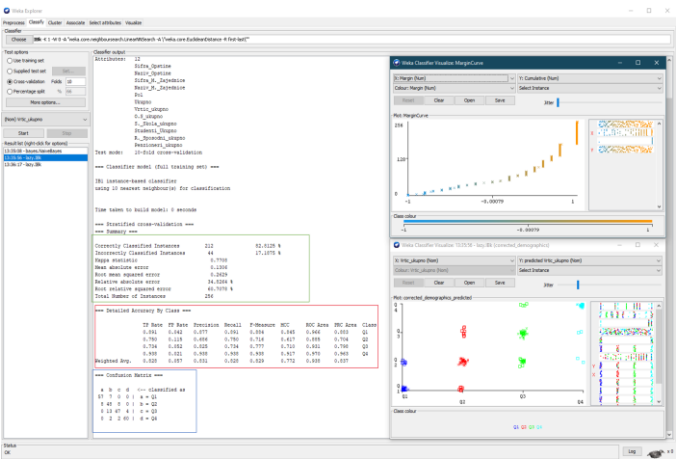

Figure 3. NN (k=10) Classification Results

The classification results obtained using the K-Nearest Neighbors (KNN) algorithm with k = 50 are presented in Fig. 4. The model was evaluated using a 10-fold cross-validation procedure on the processed demographic and education-related dataset containing 256 instances and 12 selected attributes.

The classifier correctly classified 205 instances, achieving an overall classification accuracy of 80.08%. The weighted average precision, recall, and F1-score were 0.797, 0.801, and 0.799, respectively, indicating stable predictive performance across the four analyzed demographic categories.

The confusion matrix demonstrates that most observations were assigned to the correct class, although a larger number of classification errors can be observed when compared to the KNN model with k = 10. These errors primarily occurred between neighboring categories, suggesting that increasing the neighborhood size reduced the model's sensitivity to local patterns present in the dataset.

The visualization panel illustrates the distribution of actual and predicted class labels, showing that the four target categories remain largely distinguishable. The margin curve further indicates that most instances were classified with positive confidence margins; however, the broader neighborhood consideration introduced additional smoothing effects that slightly reduced overall classification performance.

Compared with the KNN (k = 10) configuration, the KNN (k = 50) model achieved a lower classification accuracy, suggesting that excessively large neighborhood sizes may weaken the classifier's ability to capture fine-grained demographic differences. Nevertheless, the obtained results confirm that the model remains robust and provides reliable predictive performance for educational demographic analysis.

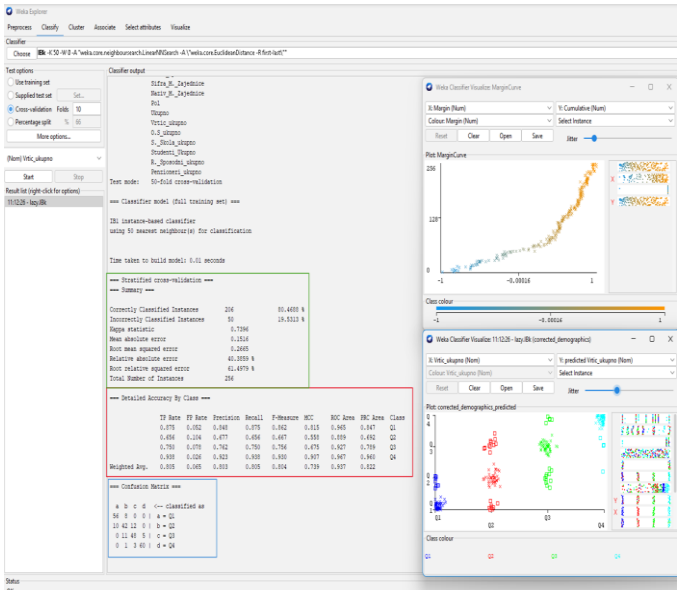

Figure 4. KNN (k=50) Classification Results

Fig. 5 presents a comparative analysis of the classification accuracy achieved by the evaluated machine learning algorithms. The comparison includes the Naive Bayes classifier, K-Nearest Neighbors with k = 10, and K-Nearest Neighbors with k = 50.

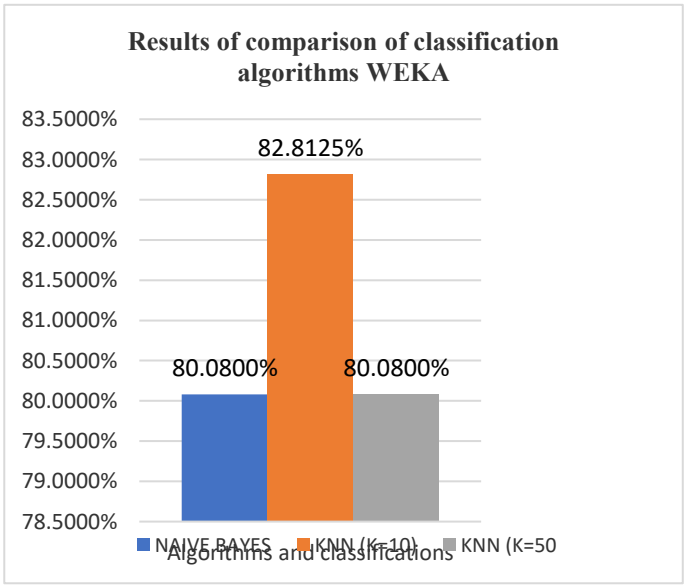

Figure 5. Classification accuracy of evaluated algorithms.

The classification results presented in Fig. 5 compare the predictive performance of the evaluated algorithms on the selected demographic and education-related dataset.

Fig. 6 presents a comparative evaluation of the Precision, Recall, and F1-score metrics obtained by the analyzed classification algorithms. These metrics provide a more comprehensive assessment of model performance than overall accuracy alone, allowing the evaluation of prediction quality, class coverage, and the balance between precision and recall.

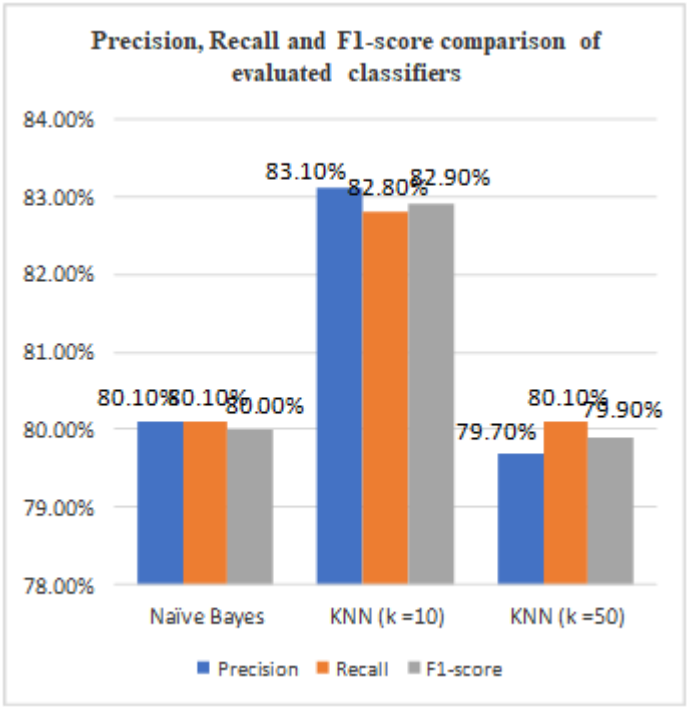

Figure 6. Precision, Recall and F1-score comparison of Naive Bayes, KNN (k = 10), and KNN (k = 50) classifiers.

Among the evaluated models, KNN with k = 10 achieved the highest overall performance, obtaining a weighted Precision of 0.831, Recall of 0.828, and F1-score of 0.829. KNN with k = 50 achieved weighted Precision, Recall, and F1-score values of 0.797, 0.801, and 0.799, respectively.

The Naive Bayes classifier achieved weighted Precision and Recall values of 0.801, with an F1-score of 0.800, demonstrating competitive performance despite its simplifying assumption of conditional independence among attributes.

The results indicate that all three classifiers achieved relatively balanced performance across the analyzed classes, as evidenced by the small differences between Precision, Recall, and F1-score values within each model. However, the KNN model with k = 10 consistently outperformed the other evaluated approaches across all performance indicators.

These findings confirm that moderate neighborhood sizes provide the most effective balance between local pattern recognition and model generalization, while also demonstrating that Naive Bayes remains a computationally efficient baseline with satisfactory predictive capabilities for educational demographic classification tasks.

Among the evaluated models, KNN with k = 10 achieved the highest classification accuracy of 82.81%, correctly classifying 212 out of 256 instances. The Naive Bayes classifier achieved an accuracy of 80.08%, while KNN with k = 50 achieved 80.08%. Although the differences between the models are relatively modest, the results indicate that neighborhood-based learning with a moderate number of nearest neighbors provides the most effective balance between local pattern recognition and generalization for the analyzed dataset.

The observed performance differences suggest that increasing the neighborhood size beyond an optimal threshold

may reduce the classifier's sensitivity to local demographic characteristics, leading to a slight decrease in predictive accuracy. Conversely, the Naive Bayes classifier demonstrated competitive performance despite its simplifying assumption of conditional independence between attributes.

Overall, the results confirm that all evaluated algorithms provide satisfactory classification performance, with KNN (k = 10) emerging as the most accurate model for the analyzed educational demographic dataset.

To complement the supervised classification experiments, an exploratory clustering analysis was conducted in Weka using the processed demographic dataset containing 256 instances and 12 attributes. SimpleKMeans and FarthestFirst were configured with three clusters (k = 3), while numerical attributes were normalized before clustering.

The SimpleKMeans algorithm converged after five iterations and generated three clusters containing 76 (30%), 83 (32%), and 97 (38%) instances, respectively. The within-cluster sum of squared errors was 1248.39. The resulting cluster distribution indicates that the processed instances were partitioned into three moderately sized groups.

The FarthestFirst algorithm produced a different cluster-size distribution, assigning 119 instances (46%) to Cluster 0, 68 instances (27%) to Cluster 1, and 69 instances (27%) to Cluster 2. This difference reflects the distinct clustering strategies of the two algorithms: SimpleKMeans iteratively refines cluster centroids to reduce within-cluster variation, whereas FarthestFirst selects distant prototype instances to establish cluster representatives.

The clustering results demonstrate that Weka supports both supervised and unsupervised exploratory analysis within a single environment. However, these clustering outputs are interpreted descriptively and as evidence of platform functionality. Since no silhouette coefficient or external cluster-validation metric was calculated, the results should not be treated as a definitive quality ranking between SimpleKMeans and FarthestFirst.

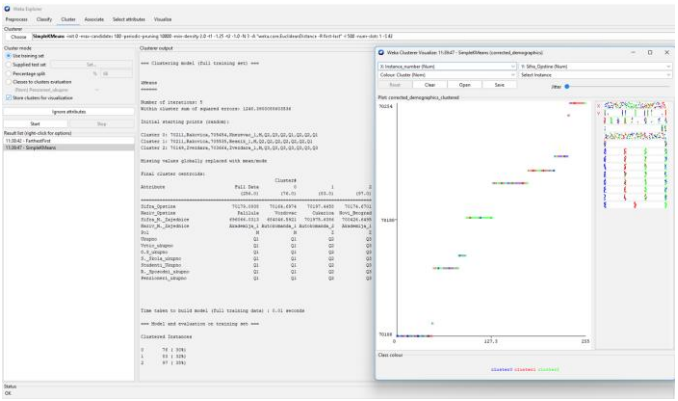

Figure 7. (a) SimpleKMeans clustering with k = 3

Fig. 7. Exploratory clustering outputs in Weka: (a) SimpleKMeans clustering with k = 3 and (b) FarthestFirst clustering with k = 3.

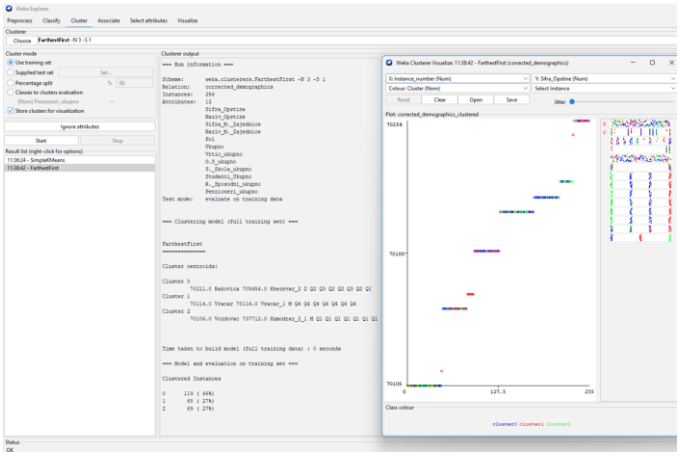

Figure 7. (b) FarthestFirst clustering with k = 3

B. *KNIME: The Pipeline Orchestrator*

KNIME excelled in the operational and orchestration aspects of the machine learning lifecycle by providing a transparent node-based environment for preprocessing, model training, evaluation, and visual interpretation of results. In contrast to WEKA's more experiment-centric interface, KNIME enabled the construction of a fully traceable analytical workflow in which each transformation and modeling step could be explicitly represented, configured, and reproduced.

As shown in **Fig. 8**, the implemented KNIME workflow begins with importing the demographic dataset from a CSV source, followed by the generation of the target class **Kategorija** through rule-based discretization of the *Vrtic ukupno* attribute into four rule-based target categories (Q1–Q4). After target generation, the workflow applies feature selection using a **Column Filter** node and performs numerical scaling through a **Normalizer** node. The prepared dataset is then divided into training and testing subsets by means of the **Table Partitioner** node, after which three parallel classification branches are executed: **Naive Bayes**, **K-Nearest Neighbors with k = 10**, and **K-Nearest Neighbors with k = 50**.

Each classification branch includes model training, prediction, and evaluation using a **Scorer** node, while additional **Bar Chart** nodes are used to visualize the distribution of predicted classes and support a more interpretable comparison of classifier behavior. This workflow-oriented design makes KNIME particularly suitable for MLOps-oriented experimentation, as it supports reproducibility, modularity, and clear documentation of the full analytical pipeline rather than only the final classification output.

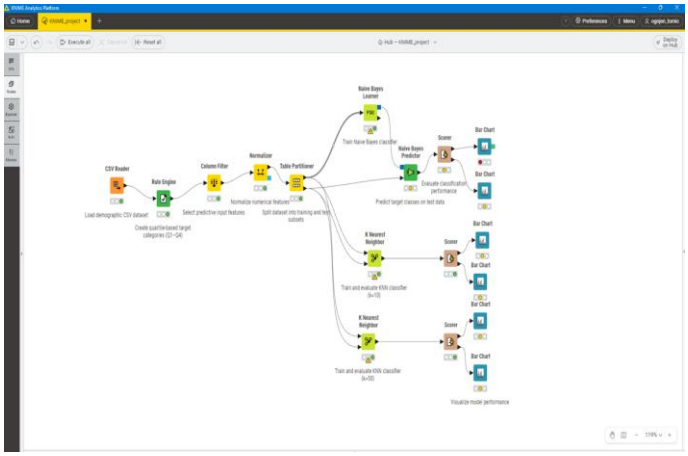

Figure 8. KNIME workflow for demographic data preprocessing, target-category generation, and comparative classification using Naive Bayes, KNN (k = 10), and KNN (k = 50).

MLOps Advantages:

- **Workflow Reproducibility:** KNIME's node-based workflows provide explicit and inspectable process documentation. Workflow versioning can be supported through repository- or Git-based practices, depending on the KNIME environment used.
- **Pipeline Automation:** Scheduling, parameterization, and workflow invocation are documented capabilities available through KNIME Server or KNIME Business Hub infrastructure [13], [15].
- **Deployment Readiness:** When combined with server- or hub-based infrastructure, KNIME workflows can be exposed as services or applications [13], [14]. These capabilities were treated as documented platform functionality and were not experimentally evaluated in this study.
- **Integration Capabilities:** KNIME provides a broad ecosystem of nodes and connectors for databases, files, cloud platforms, REST services, and enterprise systems [4].

C. NAIVE BAYES CLASSIFICATION IN KNIME

The Naive Bayes classifier was first evaluated within the KNIME workflow using the rule-based four-level target variable **Kategorija (Q1–Q4)** derived from the *Vrtic ukupno* attribute. After preprocessing, feature selection, normalization, and train–test partitioning, the classifier was trained on the training subset and evaluated on the test subset using the **Scorer** node.

The obtained results indicate **moderate classification performance**. As shown in **Fig. 9**, the Naive Bayes model correctly classified **42 instances**, while **35 instances** were misclassified, resulting in an overall **accuracy of 54.545%** and an **error rate of 45.455%**. The **Cohen's kappa coefficient of 0.29** suggests a relatively weak agreement between predicted and actual classes beyond chance, indicating that the classifier struggled to consistently distinguish between the four rule-based target categories.

A more detailed view of the class-level results reveals substantial variation in predictive performance across categories. The best-performing class was **Q2**, with a **recall of 0.848** and an **F1-score of 0.636**, indicating that most true Q2 instances were correctly identified. However, the corresponding **precision of 0.509** shows that the model frequently assigned other class instances to Q2 as well, making this category a dominant prediction target. In contrast, **Q1** was the weakest class, with a **recall of only 0.133** and an **F1-score of 0.222**, which indicates that the model failed to reliably separate this category from neighboring target categories.

The confusion matrix further confirms this pattern. A large proportion of instances from **Q1**, **Q3**, and **Q4** were incorrectly classified as **Q2**, suggesting that the feature space used in this experiment does not provide sufficiently discriminative information for Naive Bayes to cleanly separate all four target categories. While **Q4** achieved somewhat more balanced results (**precision = 0.700**, **recall = 0.500**, **F1 = 0.583**), and **Q3** reached **precision = 0.556** with **recall = 0.333**, the overall performance remains substantially below the levels observed in the WEKA experiments. This indicates that, in the KNIME hold-out evaluation setting, Naive Bayes is considerably more sensitive to class overlap and data partitioning effects than in the corresponding WEKA configuration.

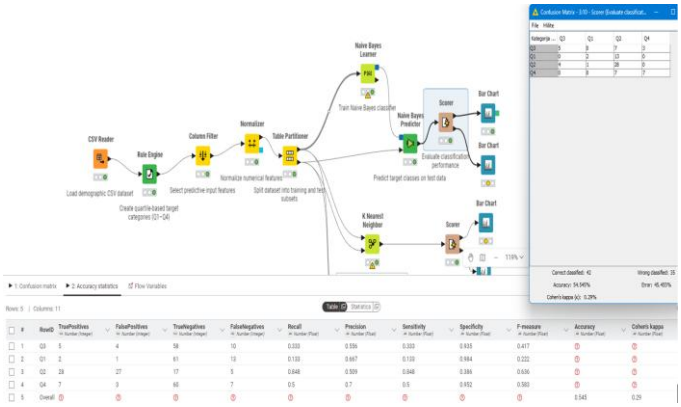

Figure 9. Naive Bayes classification results in KNIME, including confusion matrix and class-level performance metrics

D. KNN (K = 10) CLASSIFICATION IN KNIME

The second KNIME experiment evaluated an instance-based **K-Nearest Neighbors (KNN)** classifier configured with **k = 10** and the same rule-based four-level target variable **Kategorija (Q1–Q4)** used in the Naive Bayes experiment. As shown in **Fig. 10**, the model was trained and tested within the same preprocessing workflow, including target generation, feature filtering, normalization, and train–test partitioning.

The obtained results show a **substantial improvement over the Naive Bayes classifier**. The KNN model with **k = 10** correctly classified **65 instances**, while **12 instances** were misclassified, resulting in an overall **accuracy of 84.416%** and an **error rate of 15.584%**. In addition, the **Cohen's kappa coefficient of 0.769** indicates a strong level of agreement between predicted and actual class labels,

suggesting that the classifier was able to capture the structure of the rule-based demographic categories considerably more effectively than the probabilistic baseline.

The confusion matrix reveals that the strongest performance was achieved for **Q2** and **Q4**. All **33 Q2 instances** were correctly classified, indicating a highly stable decision region for this category. Similarly, **Q4** was recognized with very high reliability, with **13 correctly classified instances** and only **one misclassification**, which was assigned to **Q3**. The classifier also performed notably better on **Q1** than the Naive Bayes model, correctly identifying **10 instances**, although **5 Q1 instances** were still confused with **Q2**. For **Q3**, the classifier correctly labeled **9 instances**, while **6 instances** were incorrectly assigned to **Q2**, suggesting that some overlap remains between the Q2 and Q3 target categories.

Overall, the KNN model with **k = 10** provided the best balance between local sensitivity and classification stability among the KNIME experiments conducted so far. Compared with Naive Bayes, the improvement is evident not only in overall accuracy but also in the much clearer separation of **Q1**, **Q2**, and **Q4** categories. These results suggest that the demographic structure of the dataset is better captured through **instance-based neighborhood relationships** than through the independence assumptions imposed by Naive Bayes.

Figure 10. KNN (k = 10) classification results in KNIME, including the confusion matrix, overall accuracy, and error statistics

E. KNN (K = 50) CLASSIFICATION IN KNIME

The third KNIME experiment evaluated the **K-Nearest Neighbors (KNN)** classifier with the neighborhood size increased to **k = 50**, while keeping the same preprocessing pipeline, target variable definition, and train–test partitioning strategy used in the previous experiments. The goal of this configuration was to assess whether a larger neighborhood would provide a more stable classification boundary or, conversely, reduce sensitivity to local demographic patterns.

As shown in **Fig. 11**, the KNN model configured with **k = 50** achieved noticeably weaker performance than the **k = 10** configuration. The classifier correctly identified **42 instances**, while **35 instances** were misclassified, resulting in an overall **accuracy of 54.545%** and an **error rate of 45.455%**. The corresponding **Cohen's kappa coefficient of 0.261** indicates only a modest agreement between predicted and actual classes, suggesting that the classifier lost a substantial amount of discriminative power when the neighborhood size was increased.

The confusion matrix further confirms this behavior. The model preserved very strong recognition of the **Q2** class, correctly classifying **33 Q2 instances** with no visible confusion toward other categories. However, the performance on the remaining neighboring target categories deteriorated significantly. The **Q1** category was not correctly recognized at all, as all **15 Q1 instances** were misclassified as **Q2**. Likewise, **Q3** was heavily confused with **Q2**, with only **2 correct classifications** and **13 misclassifications into Q2**. The **Q4** class also suffered from substantial confusion, with **7 correctly classified instances**, but **5 Q4 instances** incorrectly assigned to **Q3** and **2** to **Q2**.

These results indicate that using a very large neighborhood size causes the classifier to over-smooth local class boundaries and favor the dominant regional structure represented by **Q2**. In other words, the model becomes less sensitive to localized differences between rule-based demographic categories and increasingly biases predictions toward the most centrally distributed class. Compared with **KNN (k = 10)**, the **k = 50** configuration therefore produces a substantial drop in both classification accuracy and class separability.

Overall, the KNIME experiments suggest that the **KNN classifier is highly sensitive to the choice of the neighborhood parameter**. While **k = 10** yielded the strongest performance among the evaluated KNIME models, **k = 50** significantly reduced classification quality and effectively collapsed much of the class structure into the **Q2** category. This confirms that, for the analyzed demographic dataset, a smaller neighborhood is more appropriate for capturing local patterns and preserving distinctions among the four target categories.

Figure 11. KNN (k = 50) classification results in KNIME, including the confusion matrix, overall accuracy, and error statistics

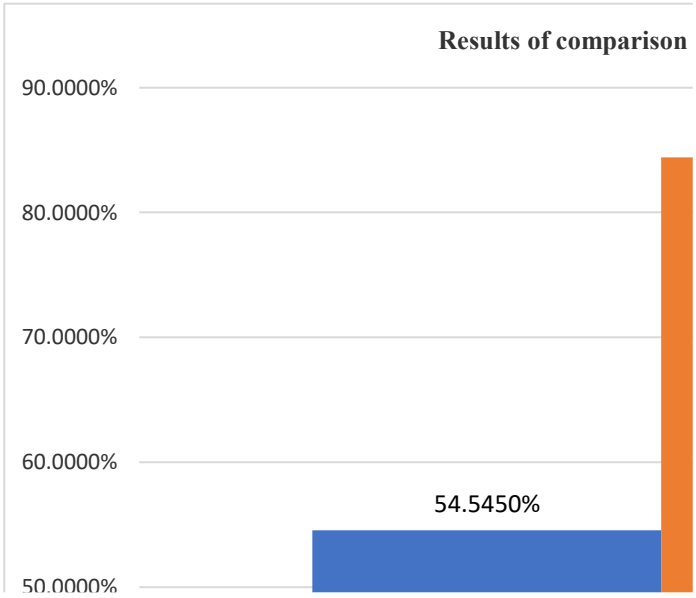

Figure 12. Classification accuracy comparison of evaluated KNIME classifiers

Fig. 12 compares the overall classification accuracy of the evaluated KNIME classifiers. Among the tested models, **KNN with k = 10** achieved the best result with **84.416% accuracy**, clearly outperforming both **Naive Bayes** and **KNN with k = 50**, which each reached **54.545%**. This finding indicates that the analyzed demographic dataset is better modeled through a local neighborhood-based approach with a moderate number of neighbors, while increasing the neighborhood size substantially reduces classification quality. The results also suggest that the Naive Bayes assumption of conditional feature independence is less suitable for this classification problem than the KNN approach with optimized parameter settings.

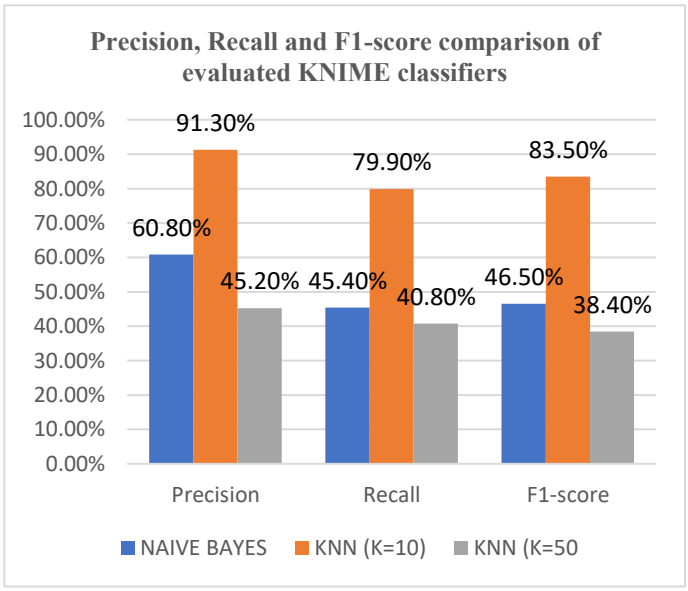

Figure 13. *Precision, Recall and* F1*-score comparison of evaluated KNIME classifiers*

Fig. 13 compares the macro-averaged **Precision, Recall, and F1-score** values obtained by the evaluated classifiers in the KNIME environment. The results confirm the same performance pattern previously observed in the accuracy comparison. Among all tested models, **KNN with k = 10** achieved the strongest overall balance between predictive correctness and class sensitivity, reaching **0.913 precision**, **0.799 recall**, and **0.835 F1-score**. These values indicate that the classifier not only produced the highest number of correct predictions overall, but also maintained the most stable class-level performance across the rule-based demographic categories.

In contrast, **Naive Bayes** produced substantially lower results, with a macro-averaged **precision of 0.608**, **recall of 0.454**, and **F1-score of 0.465**. This suggests that, although the classifier was able to correctly identify some instances of the dominant category, its overall ability to distinguish between all four rule-based target categories remained limited. The relatively low recall indicates that a considerable proportion of instances were assigned to incorrect categories, while the moderate precision reflects instability in class assignment across the dataset.

The weakest overall performance was observed for **KNN with k = 50**, which obtained **0.452 precision**, **0.408 recall**, and **0.384 F1-score**. These results show that increasing the number of neighbors substantially reduced the model's discriminative capacity and led to over-generalization of the decision boundary. In practical terms, the classifier became increasingly biased toward dominant local structures, which negatively affected both sensitivity and precision across the minority target categories.

Taken together, the values shown in Fig. 13 reinforce the conclusion that, within the KNIME workflow, **KNN with k = 10** represents the most effective classifier for the analyzed demographic categorization task. The comparison also demonstrates that model performance in KNIME is highly dependent on parameter selection, and that a larger neighborhood size does not necessarily improve generalization performance for structured demographic data.

F. Exploratory Clustering in KNIME

To complement the supervised classification workflow, an exploratory clustering branch was implemented in KNIME using the normalized demographic and education-related indicators. The clustering workflow consisted of CSV-based data ingestion, numerical feature selection, min–max normalization, clustering, aggregation of cluster profiles, and visual inspection through scatter plots and bar charts. The analysis was performed separately from the classification workflow and did not use the target category created for the supervised learning experiments.

The native k-Means implementation in KNIME was configured with three clusters. As shown in Fig. 14, the scatter plot visualizes the clustering structure using normalized primary-school and secondary-school indicators, while the accompanying bar chart summarizes the mean values of the selected demographic and education-related attributes across the three clusters. The resulting profiles indicate a differentiated distribution of instances, ranging from lower-intensity demographic and educational profiles to groups with higher normalized values across school-age, student, working-age, and pensioner indicators. This demonstrates KNIME's ability to support transparent unsupervised analysis through a modular and visually inspectable workflow.

FarthestFirst clustering was additionally executed through the Weka 3.7 integration available in KNIME, followed by cluster assignment and visual aggregation. Fig. 15 shows that the FarthestFirst procedure produced a substantially more imbalanced cluster distribution, with one dominant cluster containing most observations and two comparatively small clusters. This behavior is consistent with the prototype-selection principle of the FarthestFirst algorithm, which initializes cluster representatives using distant instances rather than iteratively minimizing within-cluster variation.

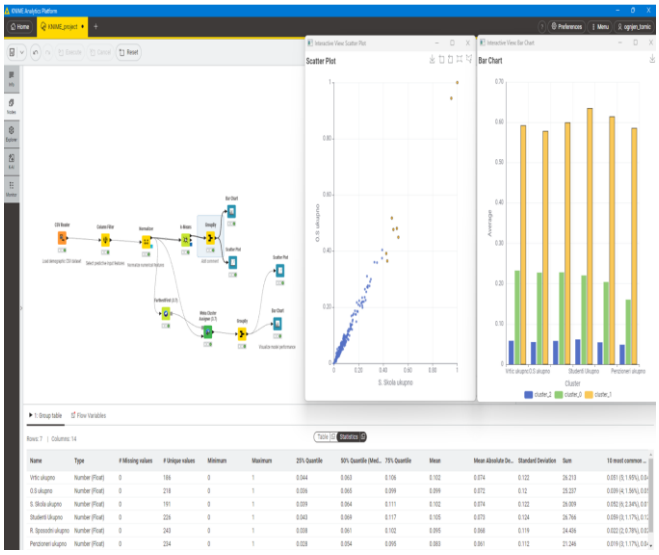

Figure 14. Exploratory k-Means clustering in KNIME using normalized demographic and education-related indicators, including scatter-plot visualization and cluster-profile comparison.

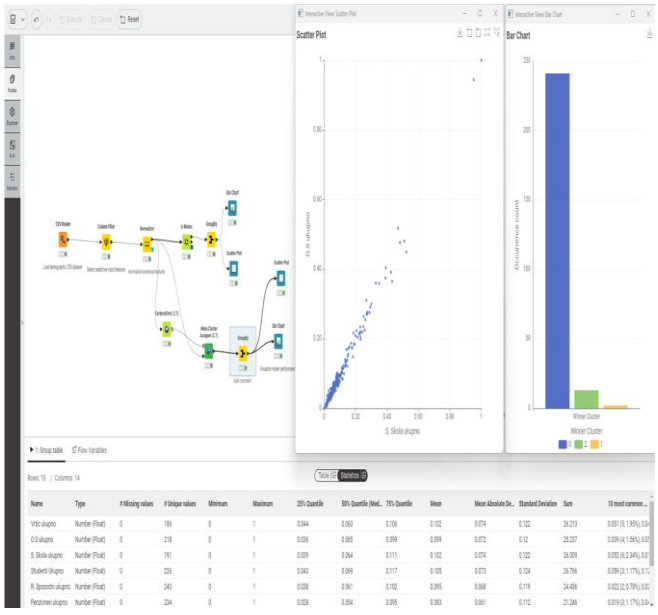

Figure 15. Exploratory FarthestFirst clustering executed through KNIME Weka 3.7 integration, showing cluster assignment and the resulting cluster-size distribution.

The clustering results are interpreted as exploratory evidence of platform functionality rather than as a definitive ranking of clustering algorithms. No silhouette coefficient or external clustering-validity metric was calculated in the present implementation. Therefore, the main contribution of this analysis is to demonstrate how KNIME can integrate data preparation, clustering, aggregation, and visualization within a reproducible workflow, complementing the more direct algorithm-oriented clustering execution available in Weka.

G. Comprehensive MLOps Capability Assessment

The detailed capability scores are presented in Table 3, the maturity scoring rubric is shown in Table 4, the aggregated maturity results are reported in Table 5, and the supporting evidence is summarized in Table 6. Together, these tables evaluate both platforms across the selected MLOps lifecycle dimensions.

The scoring does not imply that KNIME is universally superior to Weka. The evaluation is scoped specifically to MLOps lifecycle management. Weka remains stronger for lightweight algorithmic experimentation and educational prototyping, while KNIME provides stronger support for reproducible, automated and operationalizable workflows.

TABLE V. *AGGREGATED MLOPS MATURITY SCORES*

Dimension	Weka average	KNIME average	Interpretation
Workflow reproducibility	2.33	4.33	KNIME provides stronger workflow-level reproducibility.
Pipeline automation	1.67	4.00	KNIME better supports scheduled and controlled execution.
Deployment readiness	1.67	3.67	KNIME is more deployment-ready, but not fully production-observable by default.
Integration and extensibility	3.67	5.00	Both are extensible, but KNIME has broader integration coverage.
Overall maturity	2.33	4.25	KNIME better supports MLOps lifecycle management; Weka remains stronger for lightweight algorithmic exploration.

The largest differences appear in pipeline automation and deployment readiness. This outcome is expected because Weka is primarily designed as an algorithmic data mining workbench, while KNIME is designed around reusable visual workflows, automation, and integration with external systems.

TABLE VI. *EVIDENCE SUPPORTING MLOPS CAPABILITY SCORES*

Criterion	Weka evidence	KNIME evidence
Version control	Workflows and experiments can be saved, but versioning is external/manual.	Workflows can be managed as reusable artifacts and versioned through repository/Git-based practices.
Documentation	Experiment settings and results are available, but pipeline documentation is limited.	Node-based workflows provide explicit, self-documenting processing steps.
Scheduling	Limited automation; mostly manual or script-driven execution.	Scheduling is supported through KNIME Server/Business Hub infrastructure.
Model serving	Serialized models require custom Java or external integration code.	Workflows can be exposed as services/API endpoints with server/hub infrastructure.
Monitoring	No native support for model drift or production monitoring.	Execution logs and workflow monitoring exist, but advanced model observability requires additional design.
Data sources	Supports common file/database access, but with limited enterprise connectors.	Broad database, file, cloud, REST and enterprise integration ecosystem.

H. HYBRID WEKA–KNIME LIFECYCLE WORKFLOW

The proposed hybrid lifecycle consists of seven stages: (1) data ingestion; (2) data profiling and preprocessing; (3) Weka exploratory modeling; (4) candidate model selection; (5) KNIME workflow reconstruction; (6) scheduled execution, reporting, and optional API exposure; and (7) monitoring and retraining decision.

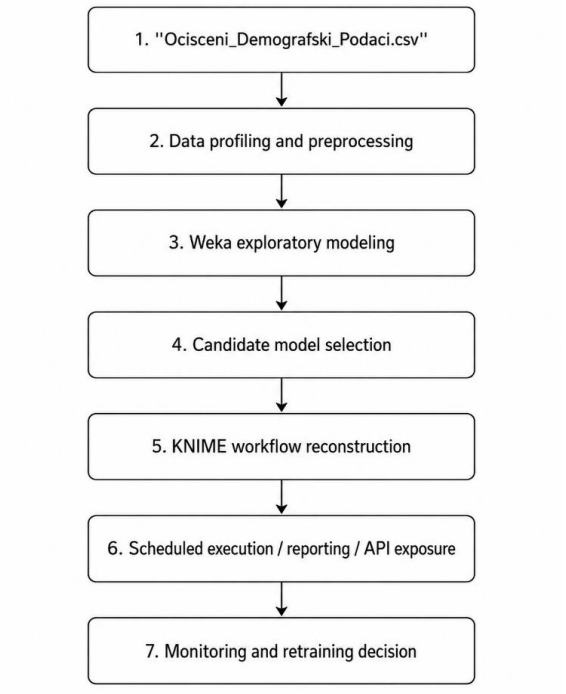

Figure 16. Fig. 16. Proposed hybrid Weka–KNIME lifecycle for MLOps-oriented Educational Data Mining.

Performance and Usability Analysis

Beyond MLOps capabilities, we observed significant differences in practical implementation:

Development Velocity:

Weka required less initial configuration for basic model experimentation, whereas KNIME required more workflow setup but provided stronger transparency, reuse, and long-term maintainability.

Maintenance Overhead:

Weka typically requires custom integration for service deployment, whereas KNIME can reduce operational overhead when supported by appropriate server- or hub-based infrastructure.

Learning Curve:

- Weka: Accessible for statisticians and researchers;
- KNIME: Requires understanding of data engineering concepts but provides greater long-term value.

Implications for Educational Institutions

The findings have significant implications for educational data mining initiatives. While Weka offers immediate analytical capabilities, KNIME provides sustainable infrastructure for ongoing data-driven decision making. This aligns with MLOps literature emphasizing reliable deployment, continuous operation, lifecycle management, and structured operational pipelines for machine learning models [5, 7, 18, 19]. The maturity-scoring framework is based on selected lifecycle dimensions and therefore may not cover all organizational, governance, and role-related aspects of MLOps maturity discussed in broader systematic reviews [17, 22].

For institutions with limited Information Technology (IT) resources, the hybrid approach we propose - using Weka for exploratory analysis and KNIME for operationalization - balances immediate needs with long-term sustainability. This strategy addresses the "deployment gap" commonly observed in educational analytics projects [12].

Practical implications for MLOps adoption in EDM

The results indicate that Weka is suitable for early-stage experimentation, algorithm selection, and teaching-oriented EDM workflows, while KNIME is more appropriate for reproducible and operationally maintainable workflows. In institutions with limited engineering resources, the hybrid workflow can reduce the gap between experimental model development and production deployment: Weka can be used to identify promising algorithms and parameter configurations, KNIME can support operationalization through scheduled execution, workflow versioning, API exposure, and integration with external data sources when server- or hub-based infrastructure is available.

V. THREATS TO VALIDITY

In the KNIME workflow, min–max normalization was applied before hold-out partitioning. Consequently, the reported KNIME metrics should be interpreted as exploratory

workflow-level results rather than leakage-free estimates of generalization performance.

Internal validity:

Maturity scores depend on evaluator interpretation. To mitigate this, scoring criteria were predefined and each workflow was implemented independently by two evaluators.

Construct validity:

The selected MLOps dimensions may not cover all production ML concerns, such as advanced observability, governance, privacy, fairness, and cost.

External validity:

External validity is limited by the use of a single demographic and education-related dataset containing approximately 15,000 records from 42 municipalities. The results may therefore not generalize to other municipalities, school systems, or educational datasets with different demographic structure, richer temporal attributes, or student-level performance data.

Conclusion validity:

Algorithmic performance results depend on preprocessing, feature selection, validation strategy, and class distribution.

Data-quality validity:

Although missing and inconsistent records were checked during preprocessing, the study did not implement a full production-grade data validation layer with automated schema checks, anomaly detection, or continuous monitoring. Such mechanisms are important in production ML pipelines and should be considered in future work [10].

Tool-version validity:

Results are tied to specific Weka and KNIME versions. Future versions may improve lifecycle support.

The study evaluates MLOps lifecycle readiness and platform capabilities in a controlled case-study setting. It does not represent a full production deployment with long-term monitoring, concept drift detection, and automated retraining in a continuously changing operational environment.

CONCLUSION

This paper presented a comparative case study of Weka and KNIME for supporting MLOps principles in Educational Data Mining. The study used the Ocisceni_Demografski_Podaci.csv dataset, containing approximately 15,000 demographic and education-related records from 42 municipalities. The reported experiments were conducted on a cleaned analytical subset of 256 instances and 12 attributes derived from the original dataset. The comparison showed that Weka is stronger in algorithm-level experimentation and rapid model testing, while KNIME is stronger in workflow-level operationalization, reproducibility, automation, integration, and deployment readiness.

A. The results do not imply that KNIME is universally superior to Weka. Rather, the findings show that the suitability of each platform depends on the phase of the machine learning lifecycle being considered. For teaching, prototyping, and small-scale offline analysis, Weka remains a practical and lightweight choice. For repeatable, auditable, and deployable EDM workflows, KNIME provides a more suitable operational foundation.

The proposed hybrid approach combines the strengths of both platforms: Weka can be used for exploratory algorithm selection, while KNIME can be used for workflow reconstruction, automation, reporting, and operationalization through server- or hub-based infrastructure. The study is limited by the use of a single dataset, the controlled evaluation setting, and the maturity-scoring framework. The reported WEKA and KNIME classifier metrics should be interpreted within their respective evaluation workflows, as WEKA was assessed through stratified 10-fold cross-validation whereas KNIME used hold-out train–test partitioning. Future work will extend the evaluation to additional educational datasets, longitudinal enrollment data, and real production deployments with monitoring and retraining mechanisms.

B. The primary contribution of this study is not the introduction of a novel machine learning algorithm, but the development and application of a practical comparative framework for evaluating traditional data mining platforms through an MLOps lifecycle perspective in Educational Data Mining.

ACKNOWLEDGMENT

This study was partially supported by the Ministry of Science, Technological Development and Innovation of the Republic of Serbia, and these results are parts of Grant No ***451-03-33/2026-03/ 200066*** *and* ***451-03-136/2025-03/200132*** *with the University of Kragujevac - Faculty of Technical Sciences Čačak.*

REFERENCES

[1] C. Romero and S. Ventura, "Educational data mining: A review of the state of the art," IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews, vol. 40, no. 6, pp. 601–618, 2010, doi: 10.1109/TSMCC.2010.2053532.

[2] R. S. J. d. Baker and K. Yacef, "The state of educational data mining in 2009: A review and future visions," Journal of Educational Data Mining, vol. 1, no. 1, pp. 3–17, 2009.

[3] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The WEKA data mining software: An update," ACM SIGKDD Explorations Newsletter, vol. 11, no. 1, pp. 10–18, 2009.

[4] A. Fillbrunn, C. Dietz, J. Pfeuffer, R. Rahn, G. A. Landrum, and M. R. Berthold, "KNIME for reproducible cross-domain analysis of life science data," Journal of Biotechnology, vol. 261, pp. 149–156, 2017, doi: 10.1016/j.jbiotec.2017.07.028.

[5] D. Kreuzberger, N. Kühl, and S. Hirschl, "Machine learning operations (MLOps): Overview, definition, and architecture," IEEE Access, vol. 11, pp. 31866–31879, 2023, doi: 10.1109/ACCESS.2023.3262138.

[6] D. Sculley et al., "Hidden technical debt in machine learning systems," in Advances in Neural Information Processing Systems, vol. 28, pp. 2503–2511, 2015.

[7] N. Hewage and D. Meedeniya, "Machine learning operations: A survey on MLOps tool support," arXiv preprint arXiv:2202.10169, 2022.

[8] A. Jović, K. Brkić, and N. Bogunović, "An overview of free software tools for general data mining," in Proceedings of the 37th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO), pp. 1112–1117, 2014.

[9] Q. A. Al-Radaideh, E. M. Al-Shawakfa, and M. I. Al-Najjar, "A comparison study between data mining tools over some classification

methods," International Journal of Advanced Computer Science and Applications, vol. 2, no. 8, pp. 18–26, 2011.

[10] E. Breck, N. Polyzotis, S. Roy, S. Whang, and M. Zinkevich, "Data validation for machine learning," in Proceedings of the 2nd SysML Conference, 2019.

[11] I. H. Witten, E. Frank, M. A. Hall, and C. J. Pal, Data Mining: Practical Machine Learning Tools and Techniques, 4th ed. Burlington, MA, USA: Morgan Kaufmann, 2016.

[12] C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell, and A. Wesslén, Experimentation in Software Engineering. Berlin, Germany: Springer, 2012.

[13] KNIME AG, "Deploy Workflows on Business Hub," KNIME Business Hub Documentation. [Online]. Available: https://docs.knime.com/hub/latest/business_hub_deployments_guide/. [Accessed: May 13, 2026].

[14] KNIME AG, "Build Your First REST Service," KNIME Community Hub Documentation. [Online]. Available: https://docs.knime.com/chub/latest/service_deployment_beginners_guide/. [Accessed: May 13, 2026].

[15] KNIME AG, "Workflow Invocation Guide," KNIME Analytics Platform Documentation. [Online]. Available: https://docs.knime.com/ap/latest/analytics_platform_workflow_invocation_guide/. [Accessed: May 13, 2026].

[16] M. Testi, M. Ballabio, E. Frontoni, G. Iannello, S. Moccia, P. Soda, and G. Vessio, "MLOps: A taxonomy and a methodology," IEEE Access, vol. 10, pp. 63606–63618, 2022, doi: 10.1109/ACCESS.2022.3181730.

[17] A. Lima, L. Monteiro, and A. Furtado, "MLOps: Practices, maturity models, roles, tools, and challenges – A systematic literature review," in Proceedings of the 24th International Conference on Enterprise Information Systems (ICEIS), vol. 1, pp. 308–320, 2022.

[18] M. Steidl, M. Felderer, and R. Ramler, "The pipeline for the continuous development of artificial intelligence models—Current state of research and practice," Journal of Systems and Software, vol. 199, Art. no. 111615, 2023, doi: 10.1016/j.jss.2023.111615.

[19] Z. Fang et al., "MLOps spanning whole machine learning life cycle: A survey," arXiv preprint arXiv:2304.07296, 2023.

[20] P. Chapman et al., CRISP-DM 1.0: Step-by-Step Data Mining Guide. Chicago, IL, USA: SPSS Inc., 2000.

[21] L. Berberi, V. Kozlov, G. Nguyen, J. Sáinz-Pardo Díaz, A. Calatrava, G. Moltó, V. Tran, and Á. López García, "Machine learning operations landscape: Platforms and tools," Artificial Intelligence Review, vol. 58, Art. no. 167, 2025, doi: 10.1007/s10462-025-11164-3.

[22] M. Zarour, H. Alzabut, and K. T. Al-Sarayreh, "MLOps best practices, challenges and maturity models: A systematic literature review," Information and Software Technology, vol. 183, Art. no. 107733, 2025, doi: 10.1016/j.infsof.2025.107733.

[23] C. Romero and S. Ventura, "Educational data mining and learning analytics: An updated survey," Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, vol. 10, no. 3, Art. no. e1355, 2020, doi: 10.1002/widm.1355.

Ognjen P. Tomić is a Ph.D. student in Information Technology at the Faculty of Technical Sciences in Čačak, University of Kragujevac, and a scientific researcher at LOLA Institute d.o.o. in Belgrade. His research interests include Machine Learning Operations, Educational Data Mining, artificial intelligence in software development, recommender systems, and data-driven decision support

Miloš Ž. Papić is affiliated with the Department of Industrial Management at the Faculty of Technical Sciences in Čačak, University of Kragujevac. His research interests include industrial management, information technologies, decision-support systems, data analytics, and the application of intelligent methods in organizational and educational environments