CloudAngles

mlangles

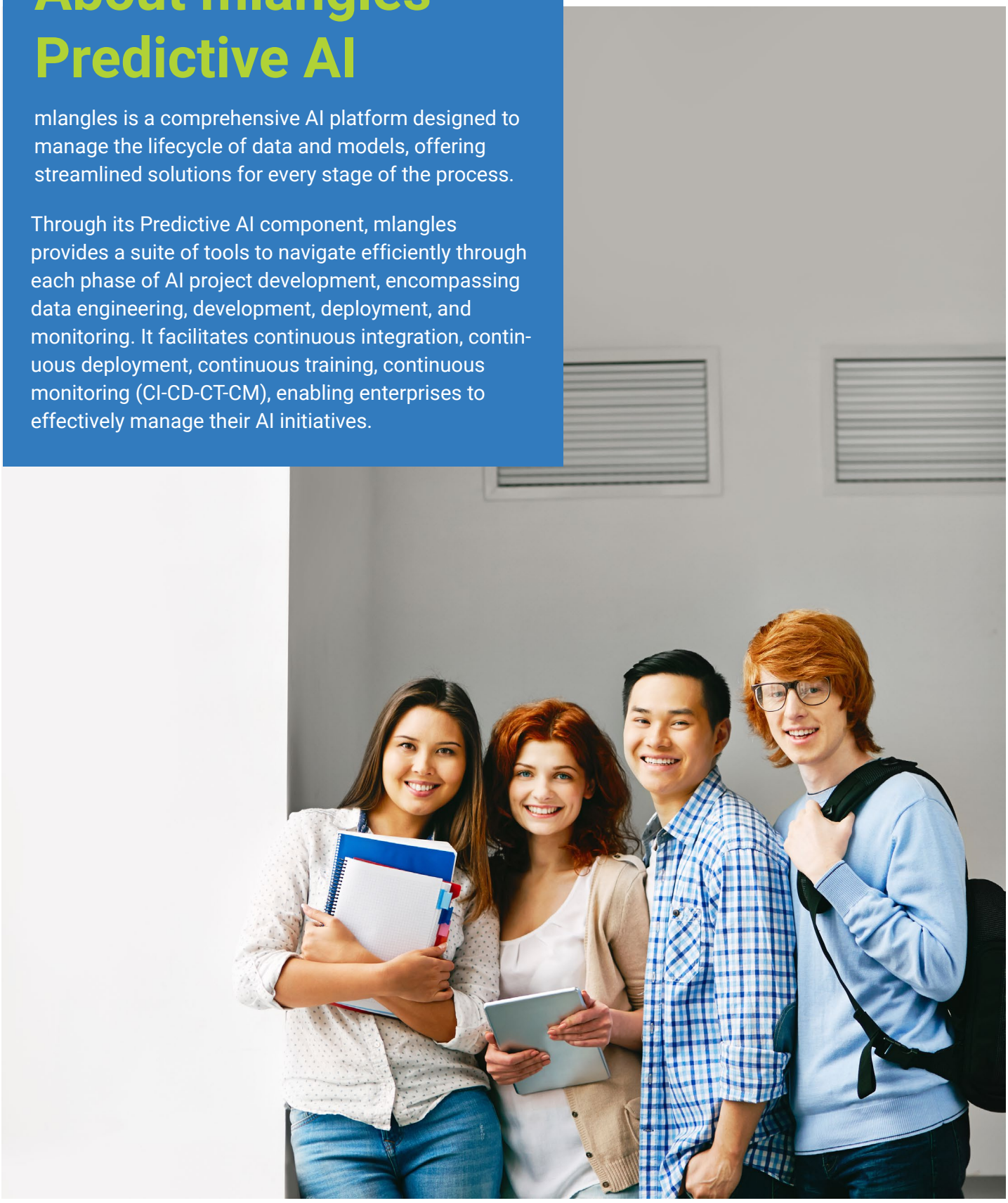
mlangles Predictive AI

Student Performance Prediction

About mlangles Predictive AI

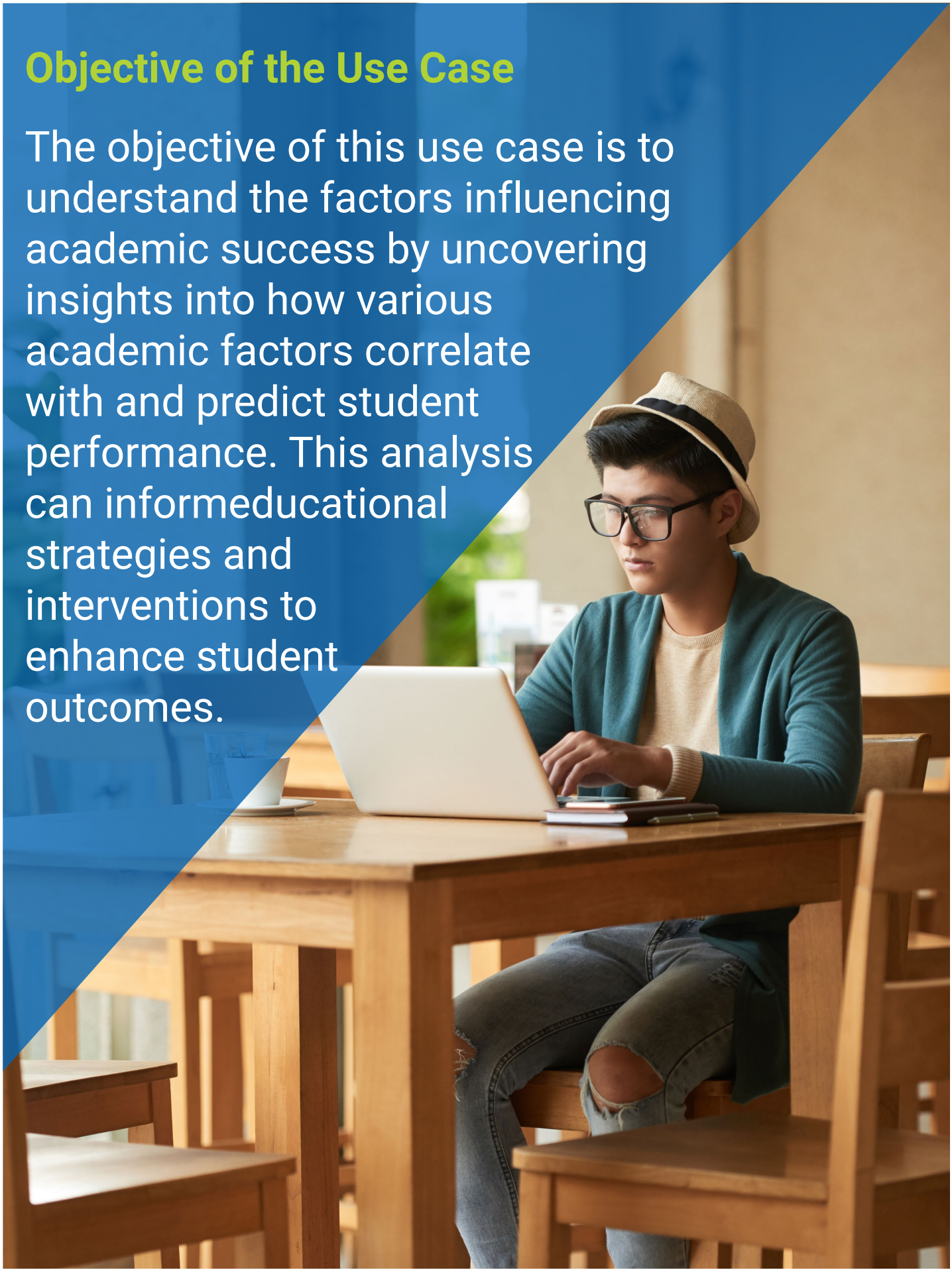
mlangles is a comprehensive AI platform designed to manage the lifecycle of data and models, offering streamlined solutions for every stage of the process.

Through its Predictive AI component, mlangles provides a suite of tools to navigate efficiently through each phase of AI project development, encompassing data engineering, development, deployment, and monitoring. It facilitates continuous integration, continuous deployment, continuous training, continuous monitoring (CI-CD-CT-CM), enabling enterprises to effectively manage their AI initiatives.



Objective of the Use Case

The objective of this use case is to understand the factors influencing academic success by uncovering insights into how various academic factors correlate with and predict student performance. This analysis can informeducational strategies and interventions to enhance student outcomes.



Explanation of the use case

In educational institutions, student performance is a critical indicator of academic success and prospects. However, understanding the factors that influence student performance can be complex. Factors such as hours studied, previous scores, extracurricular activities, sleep hours, and practice with sample question papers all contribute to a student's overall academic performance. Utilizing data-driven techniques, such as predictive analytics, educators can identify patterns and trends within student data to anticipate potential performance challenges or opportunities. By leveraging insights from these analyses, educational institutions can implement targeted interventions and support systems to optimize student outcomes and promote academic success.

The dataset consists of 10,000 student records, with each record containing information about various predictors and a performance index

Hours Studied: The total number of hours spent studying by each student.

Previous Scores: The scores obtained by students in previous tests.

Extracurricular Activities: Whether the student participates in extracurricular activities (Yes or No).

Sleep Hours: The average number of hours of sleep the student had per day.

Sample Question Papers Practiced: The number of sample question papers the student practiced.

Working of the use case

Step 1: Data Engineering & pipeline creation

Install Dependencies: The required packages and libraries are installed.

Data Extraction: The data is obtained from Kaggle, and the file is in CSV format.

Data Analysis: The following steps were performed to ensure the data integrity.

- Identify Missing Values: Check for crucial data gaps to maintain accuracy.
- Detect Duplicate Entries: Remove redundant data for consistency.

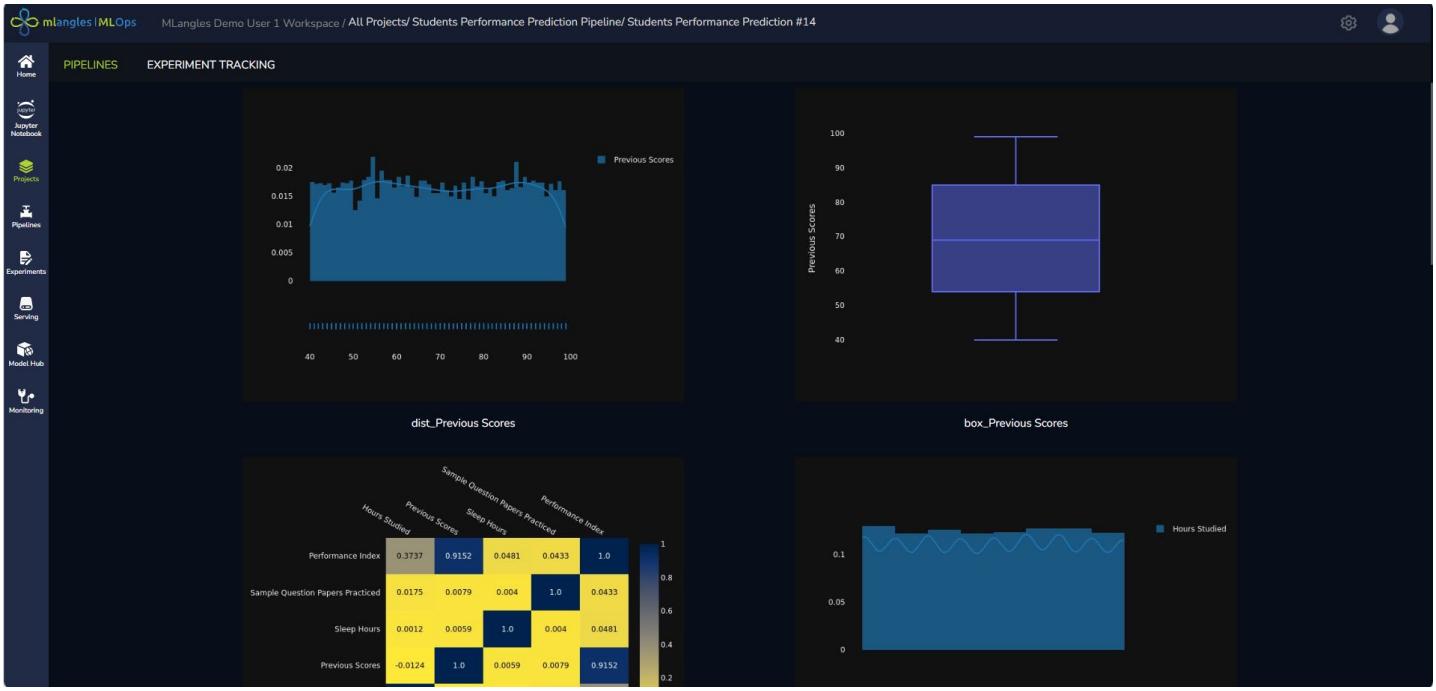
Data Preprocessing: The rows containing missing values found in data analysis are dropped and the duplicate entries are removed from the dataset

Data Visualization: This step involves representing data graphically to reveal patterns and insights. Here are brief explanations of the visualizations used in this use case.

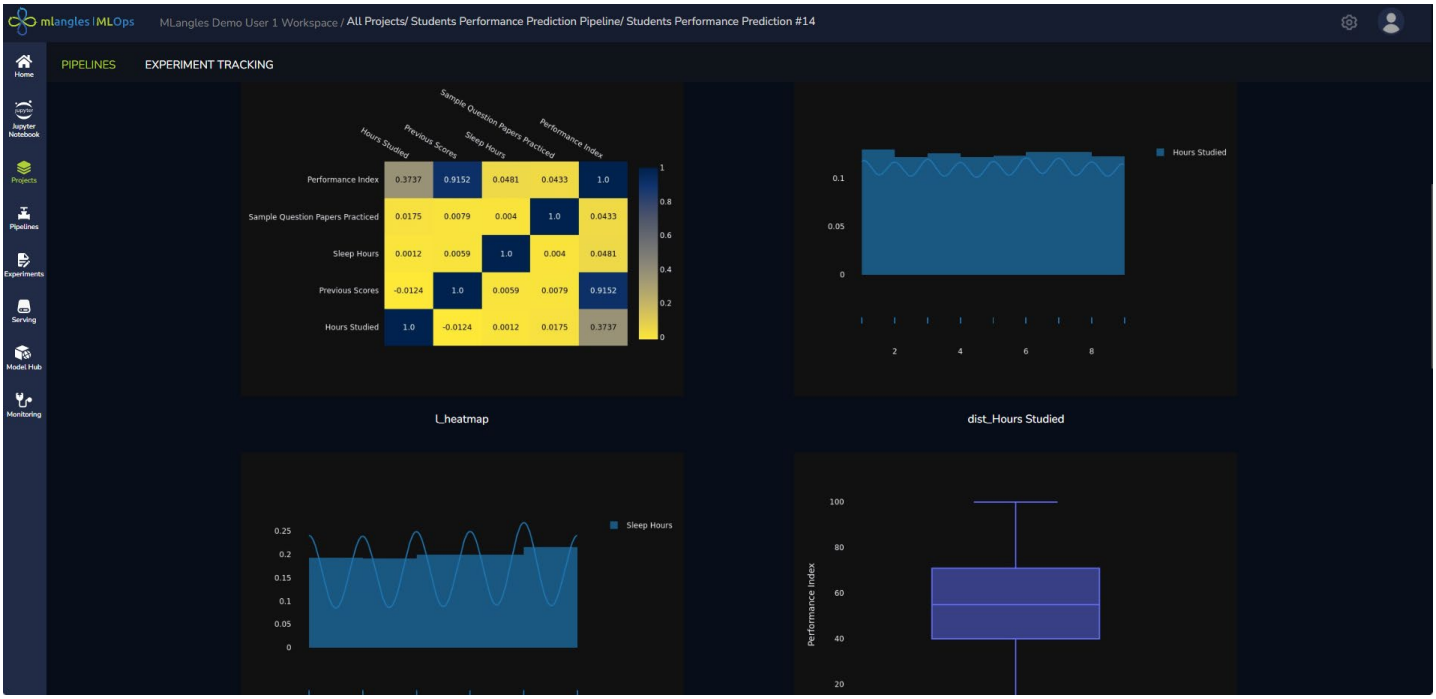
- Box Plots: Box plots display the distribution of numerical data through quartiles, highlighting the median, interquartile range, and outliers. They help identify central tendency, spread, and skewness in the data.

Heatmaps: Heatmaps visualize data in a matrix format using colors to represent values. They are useful for displaying correlations between variables or patterns in large datasets, making it easier to identify relationships and trends.

Distplots: A distplot, short for distribution plot, is a graphical representation used in statistical analysis to visualize the distribution of a dataset. It typically consists of a histogram, which displays the frequency distribution of the data, overlaid with a kernel density estimation (KDE) plot, which provides a smooth estimate of the probability density function of the data. Distplots are useful for understanding the shape, central tendency, and spread of a dataset, as well as identifying any outliers or unusual patterns The box plot below illustrates the absense of outliers within the dataset. This visualization effectively identifies potential anomalies in the data, aiding in further analysis and decision-making processes.

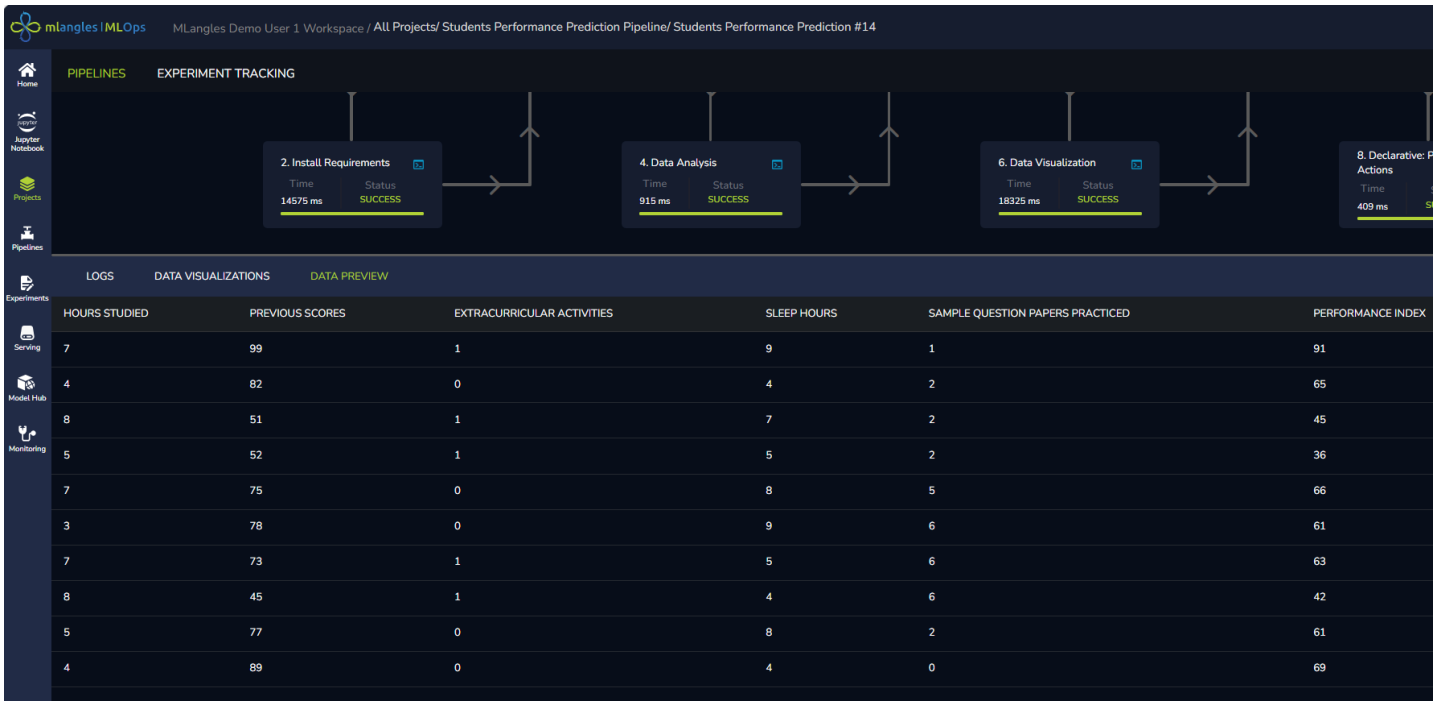


From the below heatmap, the value of previous score and performance index with a correlation coefficient of 0.9152 indicates a strong positive relationship between these two features. This suggests a higher previous score will result in a higher performance index. In other words, students who have achieved higher scores in previous tests are likely to perform better overall.



Feature Engineering: This step encompasses various tasks aimed at enhancing the quality and relevance of features used in machine learning models. This involves encoding the features having categorical variables into numerical values. Here the 'Extracurricular Activities' feature consists of yes/no values which are converted to 1/0 respectively. Additionally, removing outliers improves the robustness and generalization capability of the model by reducing the influence of anomalous data points that may distort the learning process. Overall, feature engineering enhances the quality of input features, leading to more accurate and reliable machine learning models.

Below is the image of the cleansed dataset after the data engineering steps.



HOURS STUDIED	PREVIOUS SCORES	EXTRACURRICULAR ACTIVITIES	SLEEP HOURS	SAMPLE QUESTION PAPERS PRACTICED	PERFORMANCE INDEX
7	99	1	9	1	91
4	82	0	4	2	65
8	51	1	7	2	45
5	52	1	5	2	36
7	75	0	8	5	66
3	78	0	9	6	61
7	73	1	5	6	63
8	45	1	4	6	42
5	77	0	8	2	61
4	89	0	4	0	69

Data Versioning:

- Various processed data versions can be generated through different transformations applied to the same raw dataset, such as deleting columns or applying various transformations on specific columns.
- Throughout the data pipeline, diverse transformations can be executed at each iteration. Consequently, the resulting data at the pipeline's end is systematically versioned.
- Given that each version of the final data is distinct, models trained on these different versions will exhibit varying behaviors.

Step 2: Experiment Tracking - Modelling with Hyper-Parameter Optimization

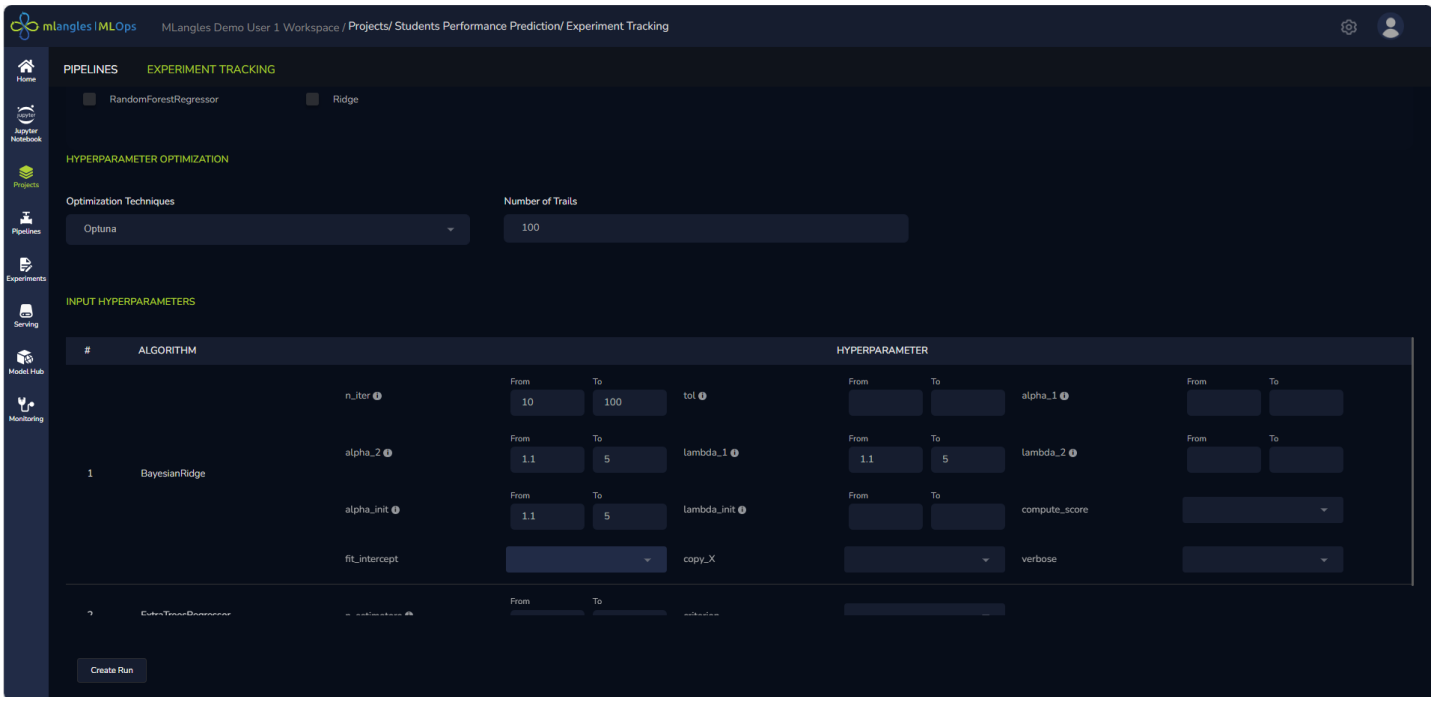
After preparing the cleansed data, the next step involves training the model using this refined dataset. Given that this is a regression problem, various models are suitable for the task. Common choices include the random forest regressor, extra trees regressor, ridge regressor, and Bayesian regressor.

Random forest regressor: It is a machine learning algorithm that builds multiple decision trees during training and outputs the average prediction of the individual trees. It offers robustness against overfitting and the ability to handle large datasets with high dimensionality.

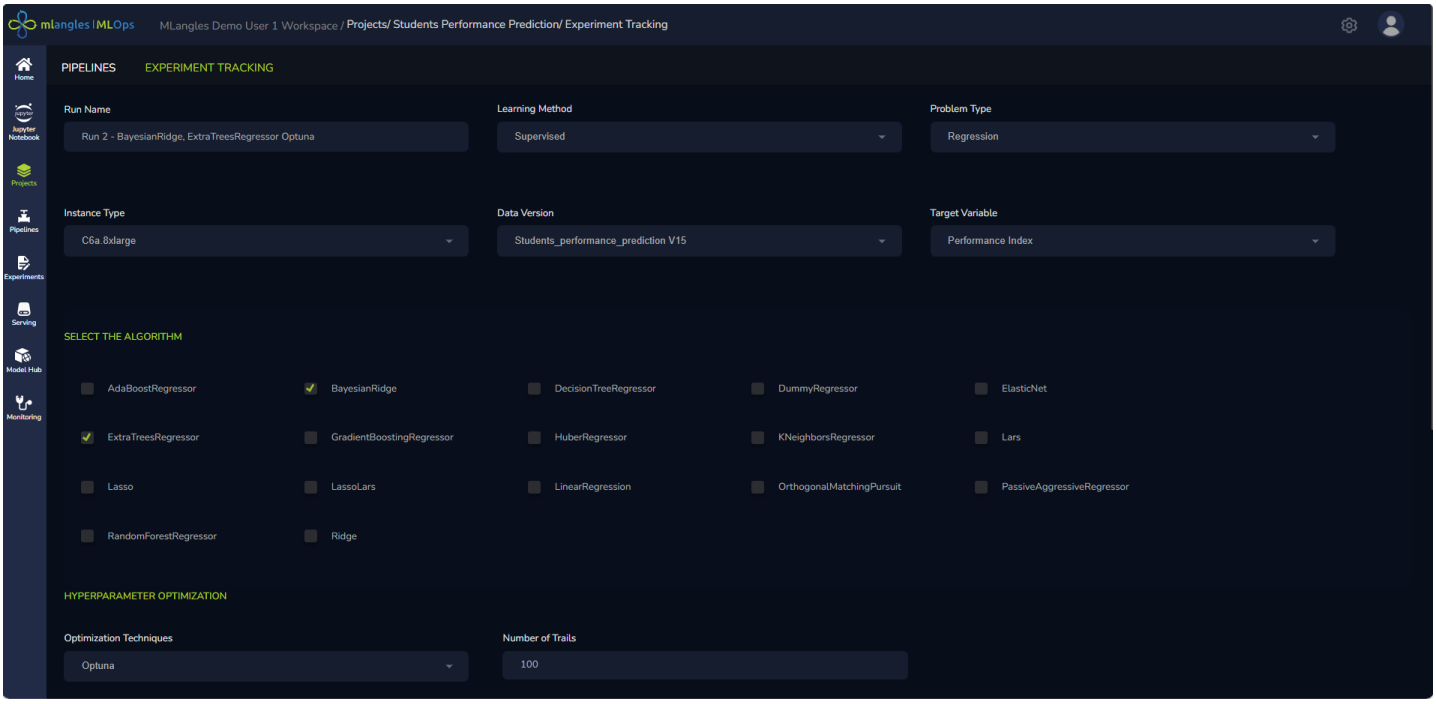
Extra trees regressor: It constructs multiple decision trees based on random subsets of features and random splits to make predictions. However, Extra Trees Regressor further randomizes the tree-building process by selecting random thresholds for splitting, leading to faster training times.

Ridge regressor: It is a linear regression algorithm that incorporates regularization to prevent overfitting. It adds a penalty term to the ordinary least squares objective function, which penalizes large coefficients. This penalty term, controlled by a hyperparameter called the regularization strength or alpha, helps to reduce model complexity and mitigate the effects of multicollinearity in the dataset.

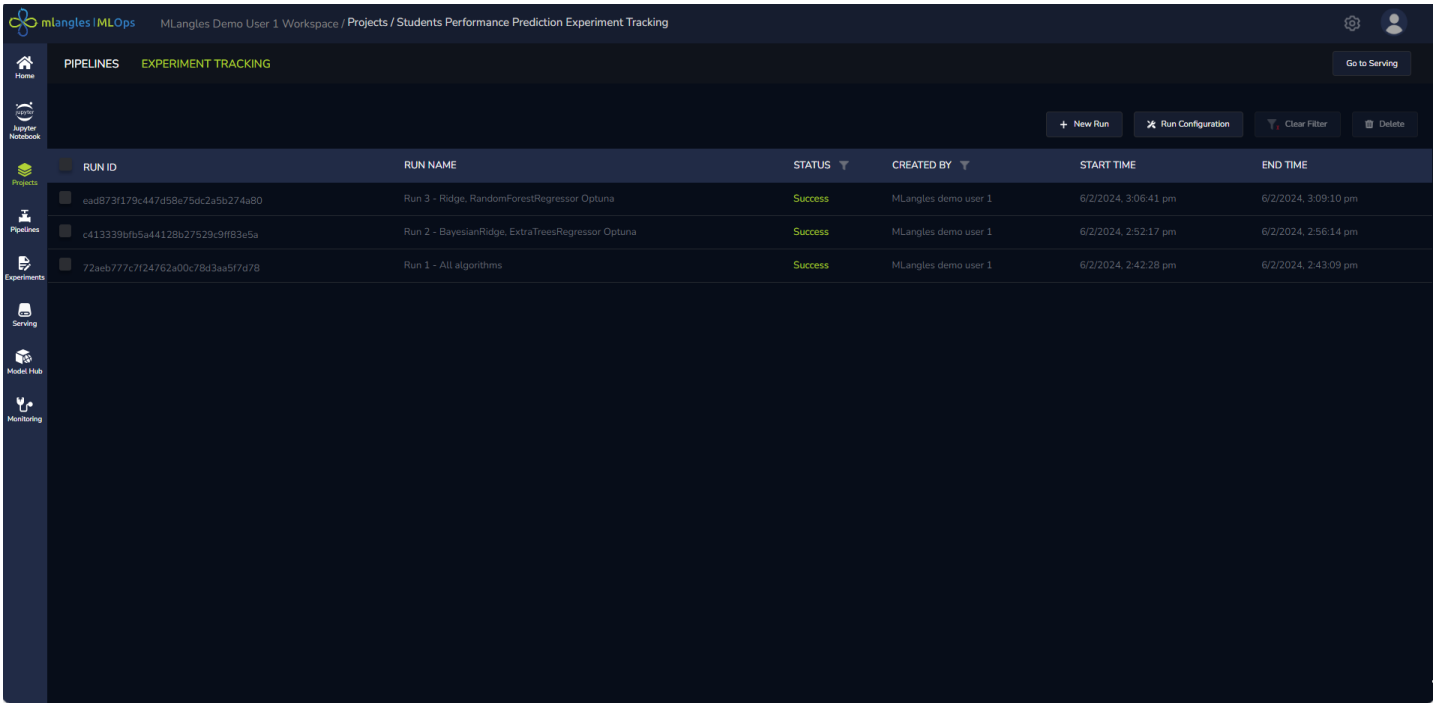
Bayesian regressor: The Bayesian Ridge Regressor is a linear regression algorithm that applies Bayesian methods to estimate model parameters. Unlike traditional linear regression, which estimates parameters using point estimates, Bayesian regression treats model parameters as random variables with prior distributions. It updates these distributions based on observed data to obtain posterior distributions, providing a more robust estimation of uncertainty.



Additionally, to enhance model performance, a hyperparameter optimization technique called Optuna is employed. Optuna automates the process of tuning hyperparameters, such as learning rate or tree depth, to find the optimal configuration that maximizes model performance. This approach ensures that the model is fine-tuned to achieve the best possible results on the given dataset, improving its accuracy and predictive power.



Once the run is created, various details can be extracted, including hyperparameter visualizations of the best algorithm, parameters utilized during training, and metrics and artifacts. These insights provide valuable information about the model's performance and behavior, aiding in understanding its effectiveness and potential areas for improvement.

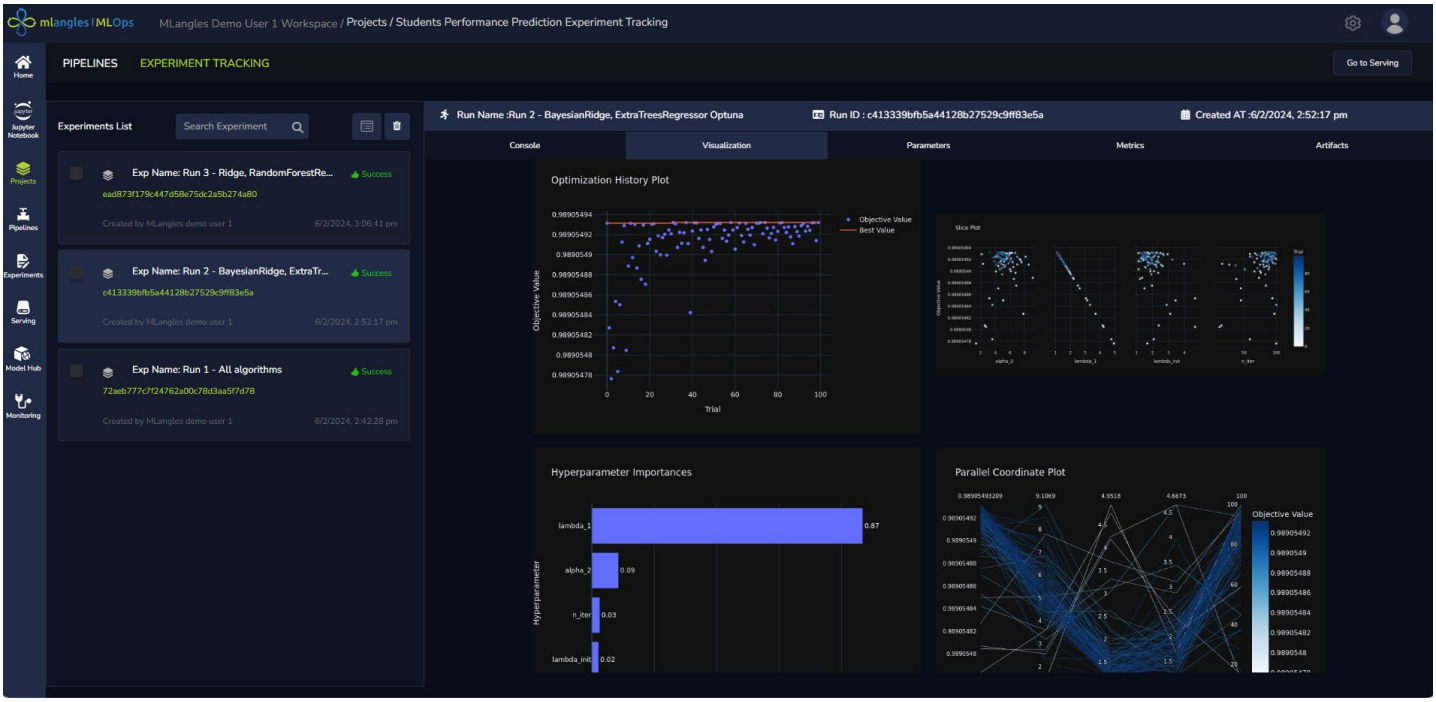


Hyperparameter visualizations offer a graphical representation of how different parameter settings impact model performance, facilitating the selection of optimal configurations.

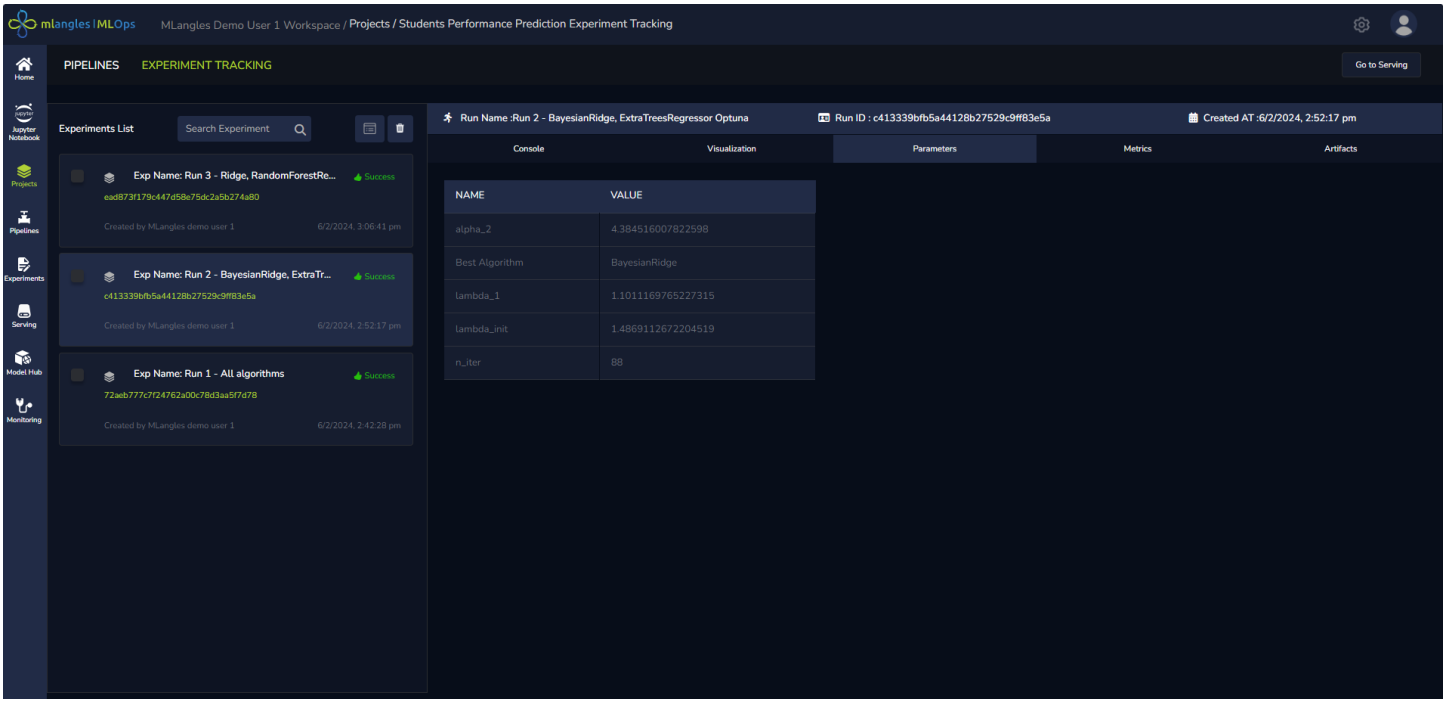
Slice Plot: A slice plot visualizes the relationship between two hyperparameters while fixing the values of other hyperparameters. It allows for the examination of interactions between hyperparameters and their effects on model performance, helping in identifying optimal parameter combinations.

Hyperparameter Importances Plot: This plot ranks the importance of hyperparameters based on their influence on model performance. It helps in identifying the most influential hyperparameters, guiding further optimization efforts or feature selection strategies. combinations.

Parallel Coordinate Plot: This plot visualizes high-dimensional hyperparameter spaces by representing each hyperparameter as a vertical axis and each point in the plot as a hyperparameter configuration. Lines connecting points represent hyperparameter configurations, enabling the exploration of relationships and patterns across multiple hyperparameters simultaneously.



Hyperparameter visualizations offer a graphical representation of how different parameter settings impact model performance, facilitating the selection of optimal configurations.



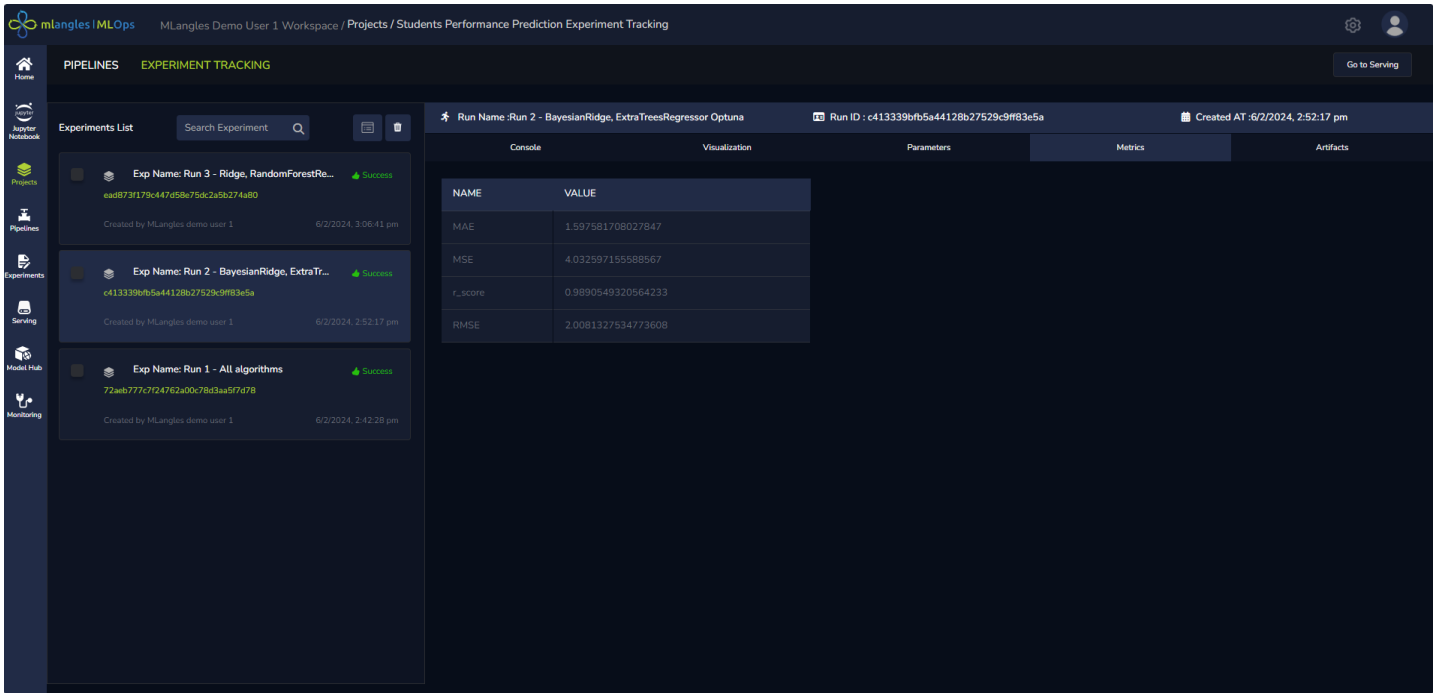
These are common evaluation metrics used in regression tasks to assess the performance of predictive models:

Mean Absolute Error (MAE): It calculates the average absolute differences between the predicted values and the actual values. MAE is less sensitive to outliers compared to other metrics like MSE.

Mean Squared Error (MSE): It calculates the average squared differences between the predicted values and the actual values. MSE penalizes larger errors more heavily than MAE and is sensitive to outliers.

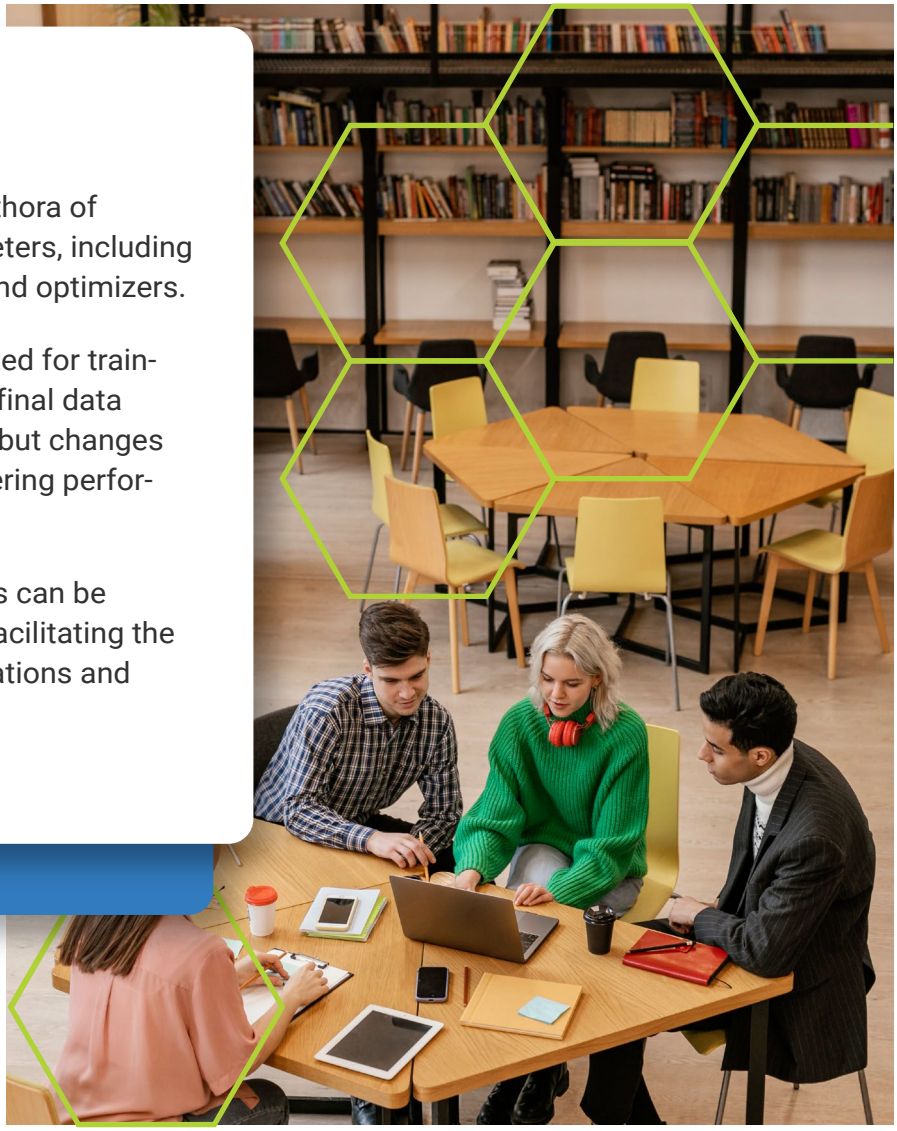
R-squared (R2 score): It measures the proportion of the variance in the dependent variable (target) that is predictable from the independent variables (features). R2 score ranges from 0 to 1, with higher values indicating a better fit of the model to the data.

Root Mean Squared Error (RMSE): It is the square root of the MSE, providing an interpretable measure of the average magnitude of errors. RMSE is in the same unit as the target variable, making it easy to understand in context.



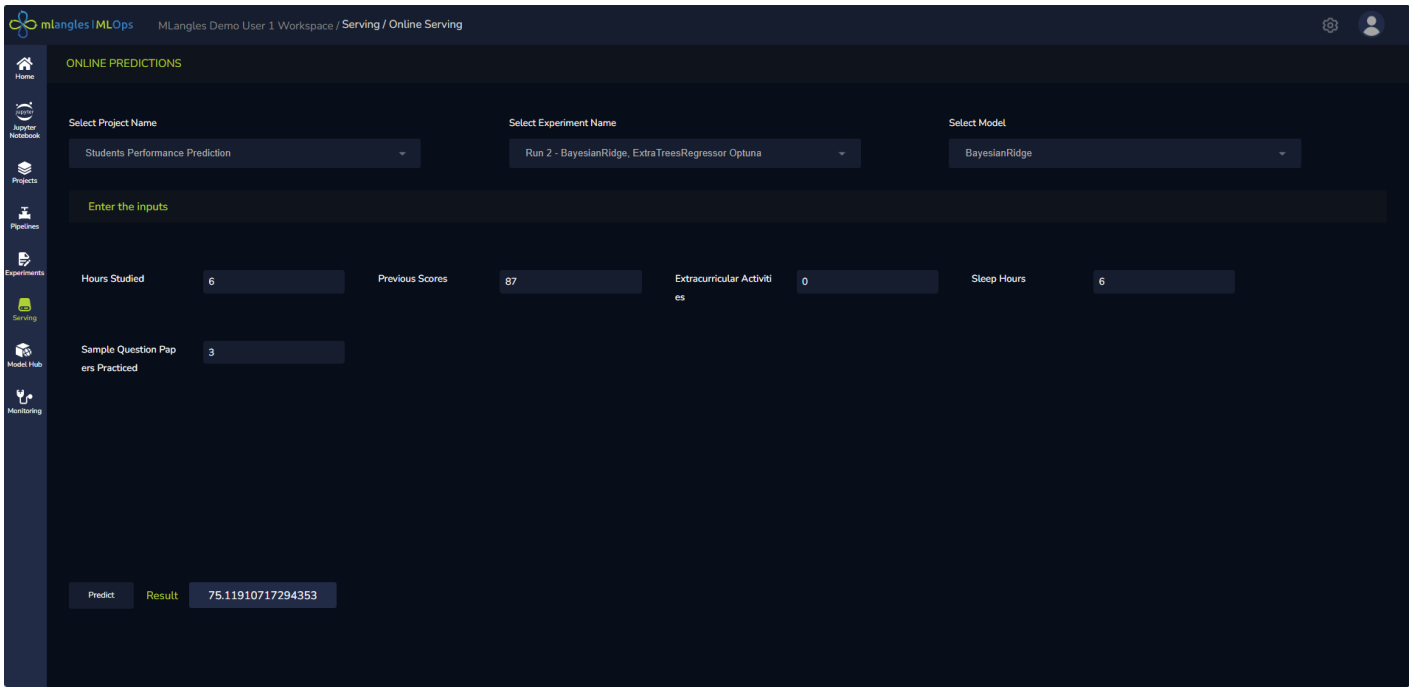
Model Versioning:

- Models are sensitive to a plethora of hyperparameters and parameters, including learning rate, loss function, and optimizers.
- Consequently, a model selected for training, with both the model and final data versions remaining constant but changes in parameters, may yield differing performance metrics.
- These diverse model versions can be uploaded to the model hub, facilitating the management of multiple iterations and variations.



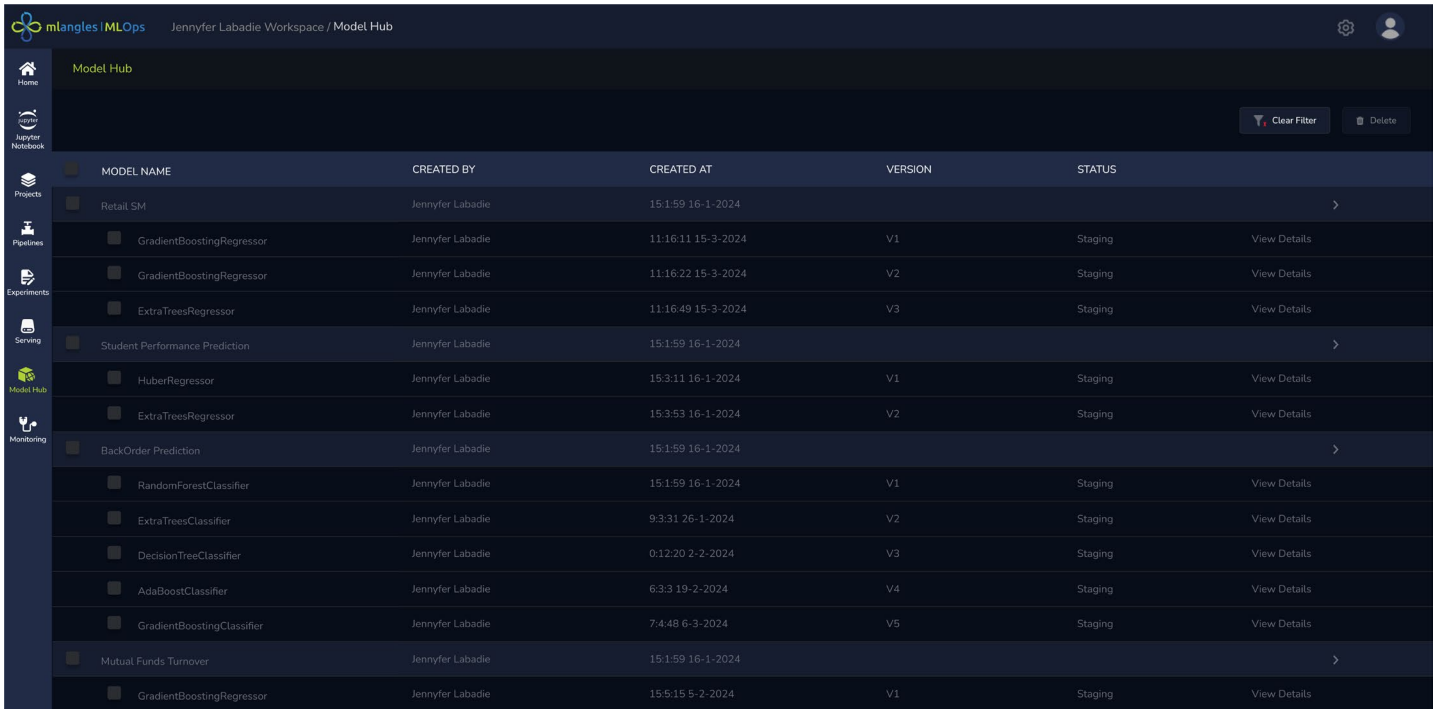
Step 3: Prediction / Serving

During model serving, a single data point is used as a sanity test for the model, and the output is predicted on a scale of 0-100 which describes the performance index of the student with the given input parameters.



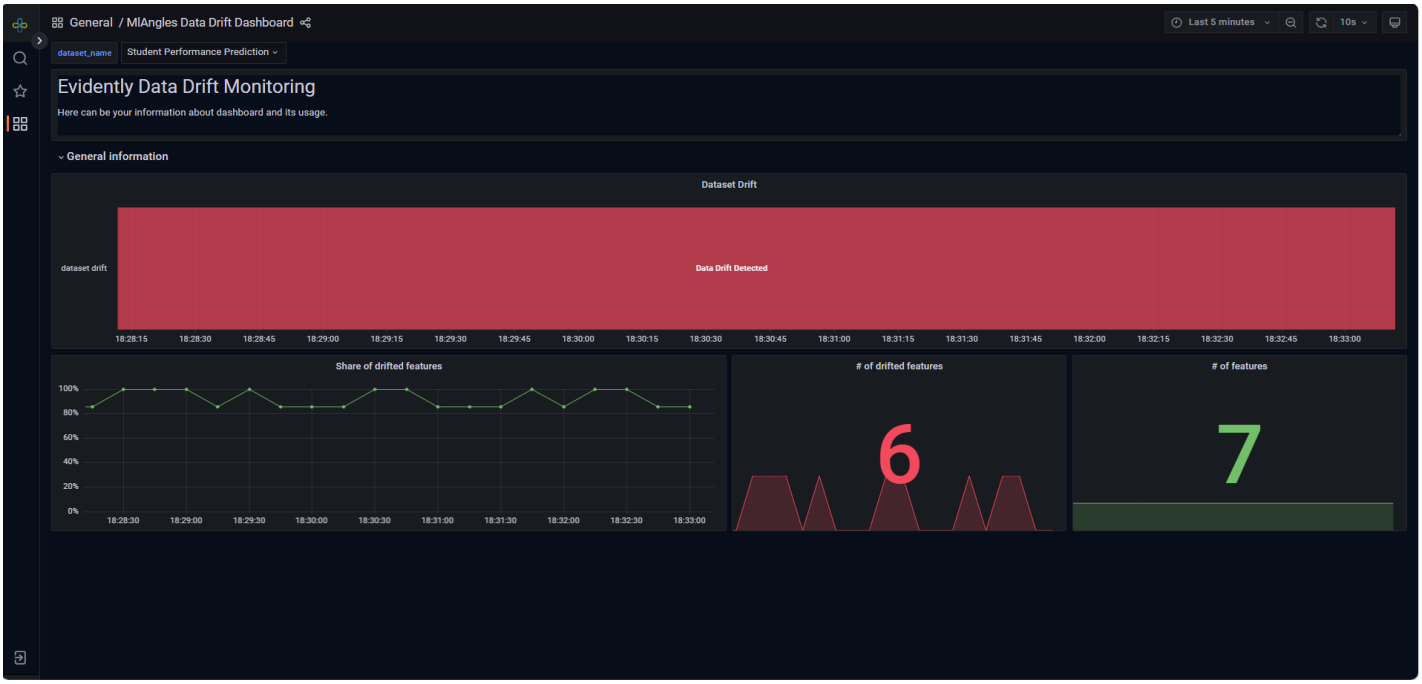
Model Hub:

- Trained models are uploaded to the model hub, whereupon deployment, a REST API endpoint is automatically generated.
- Data is transmitted to this endpoint as a request, triggering the model to execute a prediction and return the output as the response to the request.



Step 4: Monitoring

Data drift refers to the phenomenon where the statistical properties of the data change over time in a deployed machine learning model. This could be due to changes in the underlying data distribution, data collection process, or external factors influencing the data. When data drift occurs, the relationships between features and the target variable may change, impacting the model's performance and reliability. The below screen shows that there is drift in the data and 6 out of 7 features in the dataset have drifted



The share of drifted features refers to the proportion of features in the dataset that have experienced a significant change or drift in their statistical properties. As these features undergo drift, their relationships with the target variable may become less relevant or even misleading, leading to decreased model accuracy and effectiveness. Therefore, monitoring and addressing data drift are essential to maintain the model's performance and ensure its continued relevance in production environments.

Conclusion

This project aimed to predict students' performance index using the Student Performance Dataset, investigating the relationships between variables like studying hours, previous scores, extracurricular activities, sleep hours, and sample question papers practiced. The analysis provides valuable insights into the factors influencing academic performance, which can inform educational strategies and interventions aimed at improving student outcomes. Further research and analysis could explore additional variables or refine modeling techniques to enhance predictive accuracy and robustness. Overall, this project contributes to the ongoing efforts to understand and support student success in educational settings.