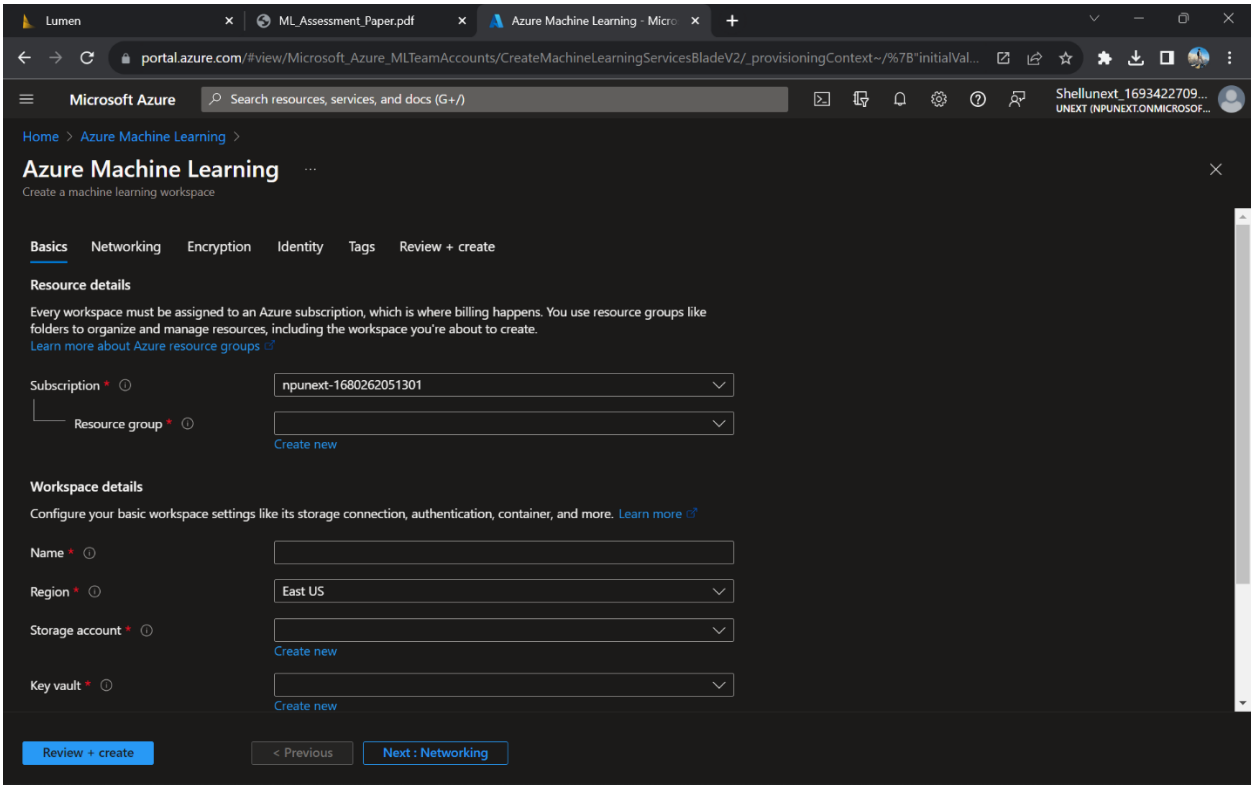
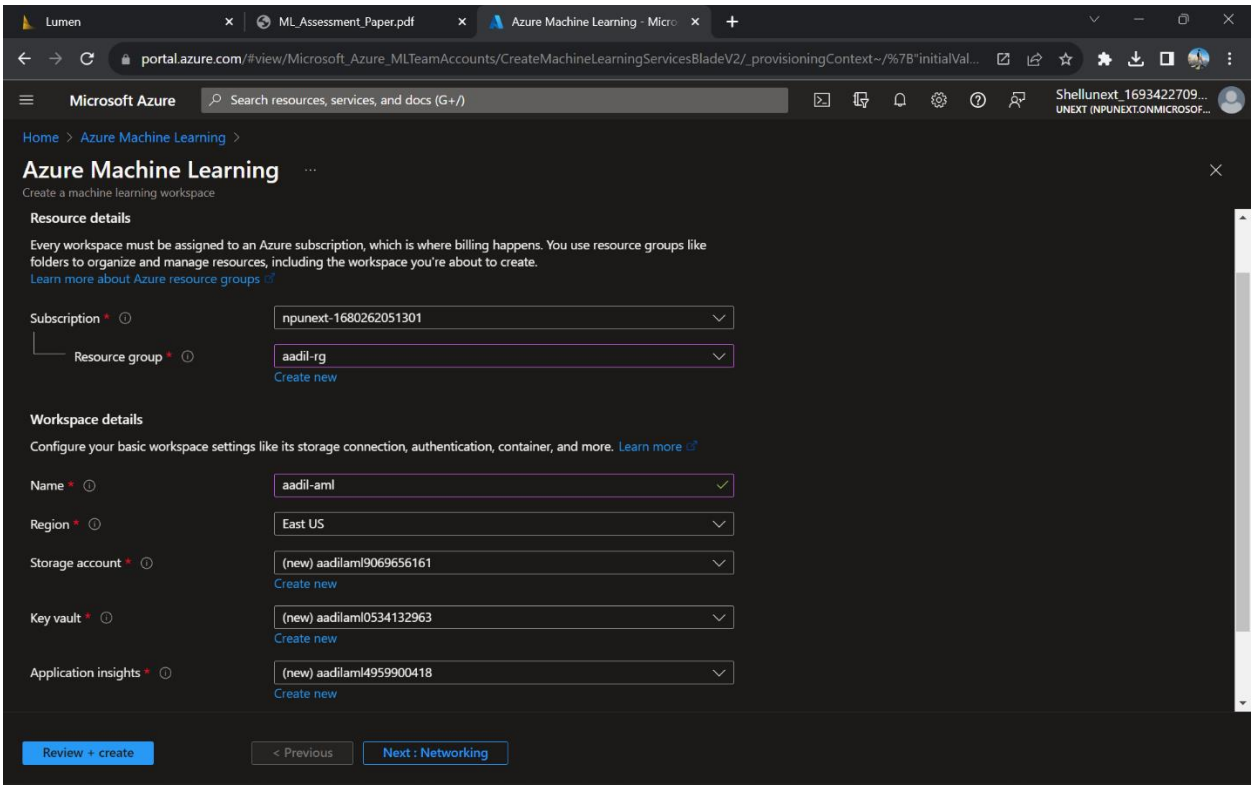
Azure Machine Learning – Hands On

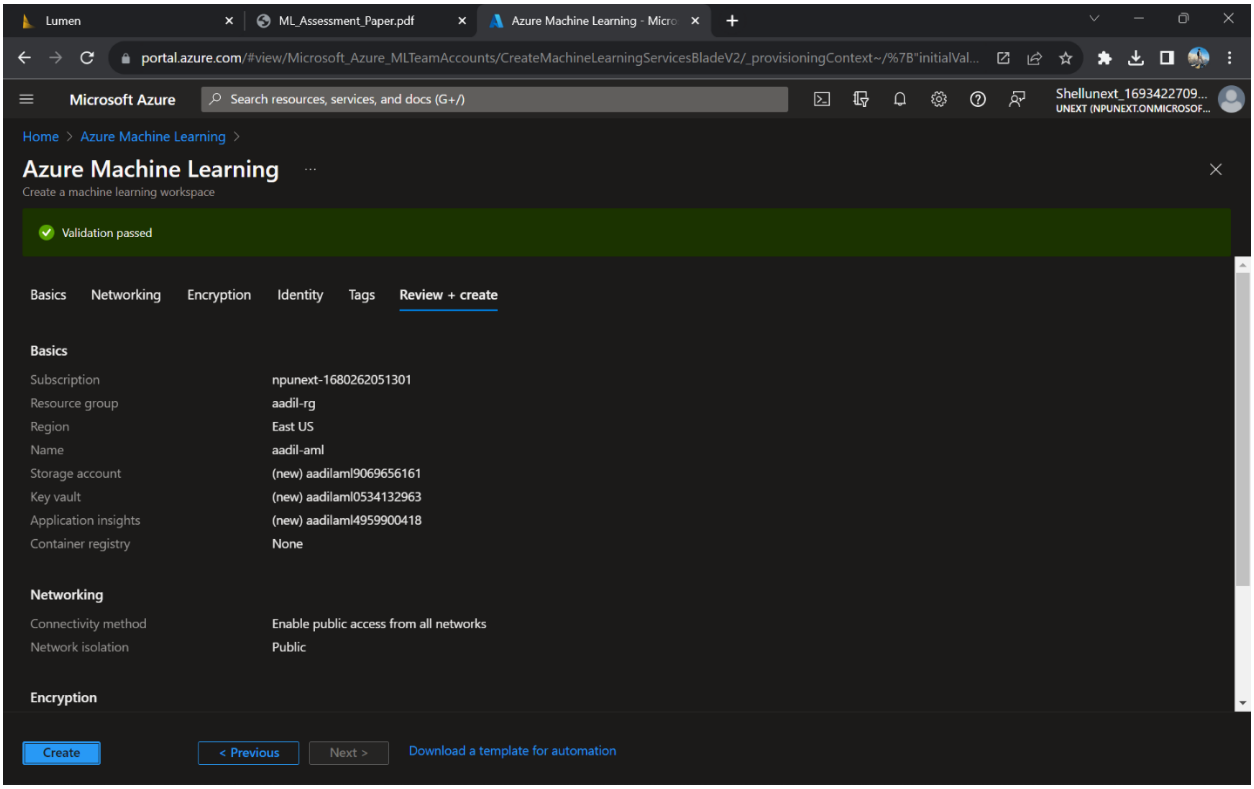
Mohammad Aadil Sehrawat
Employee Code: 654985



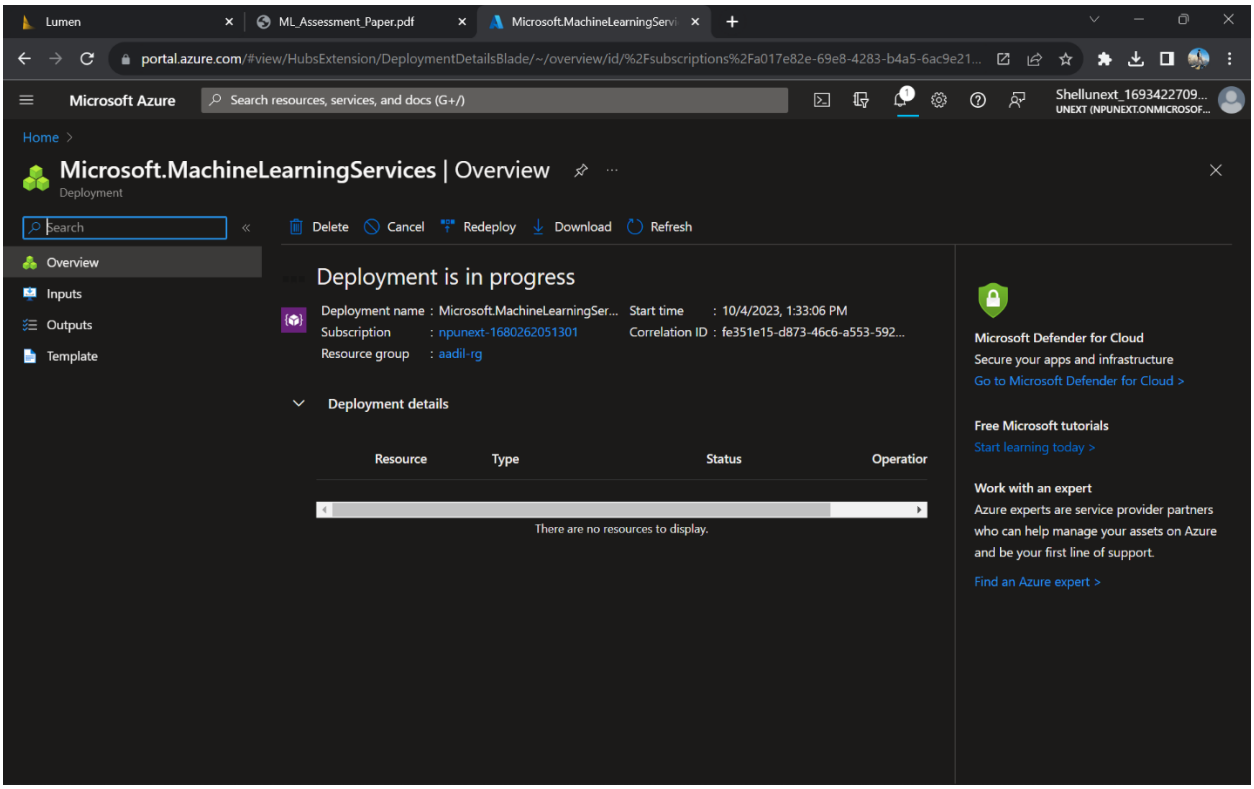
Creating a new Azure Machine Learning Instance



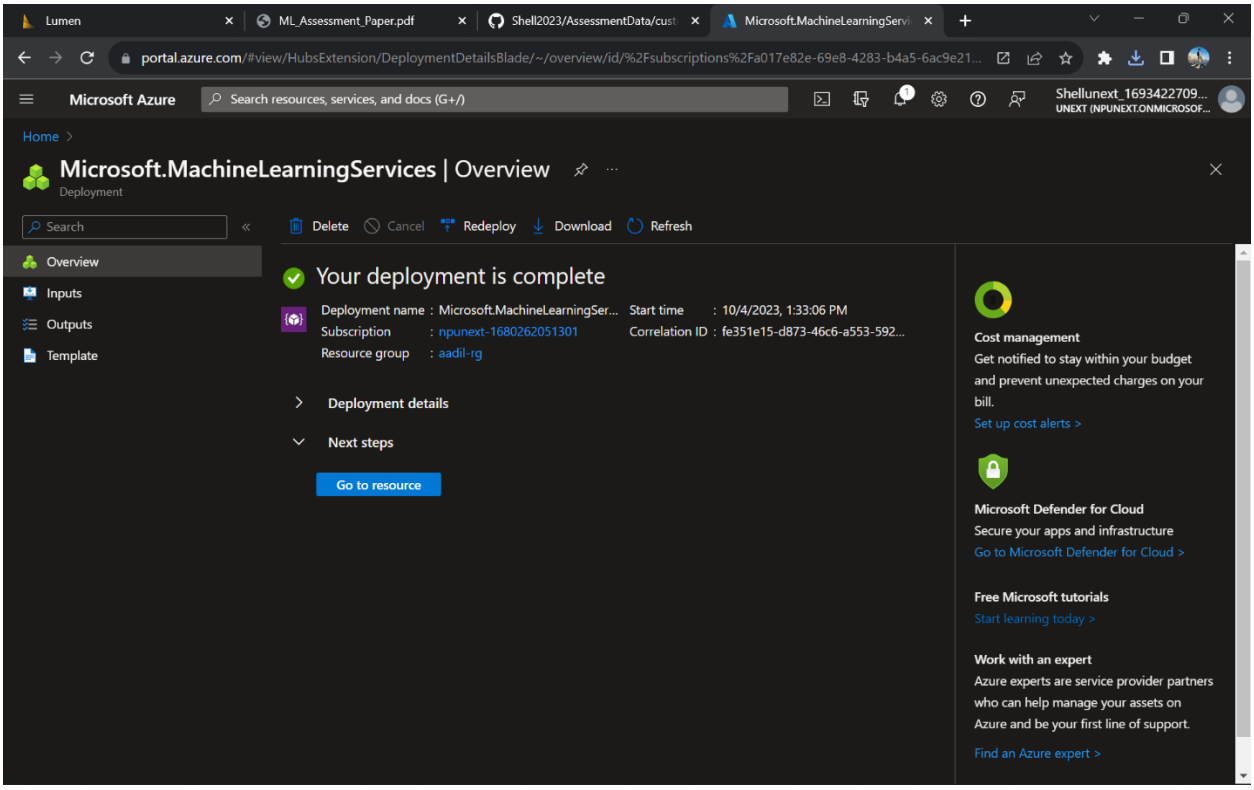
Configuration for new Azure Machine Learning Instance



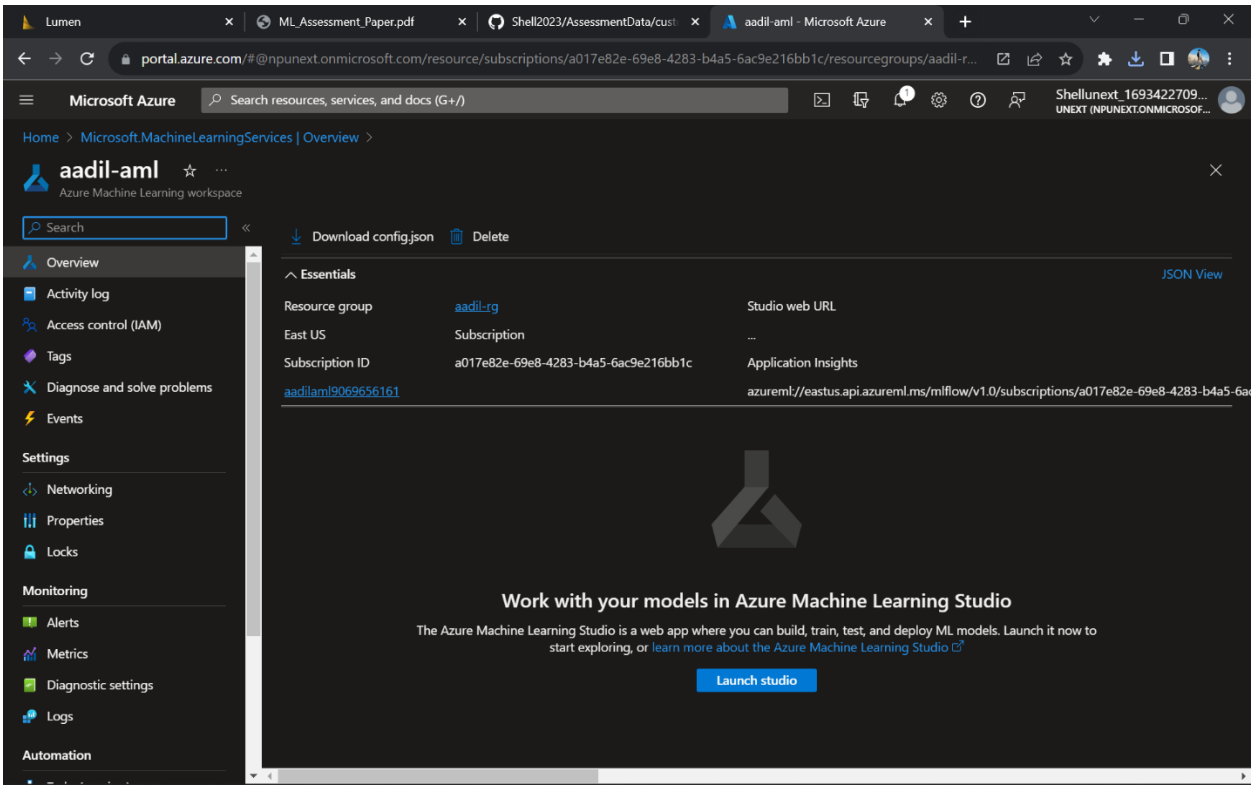
Creation under progress



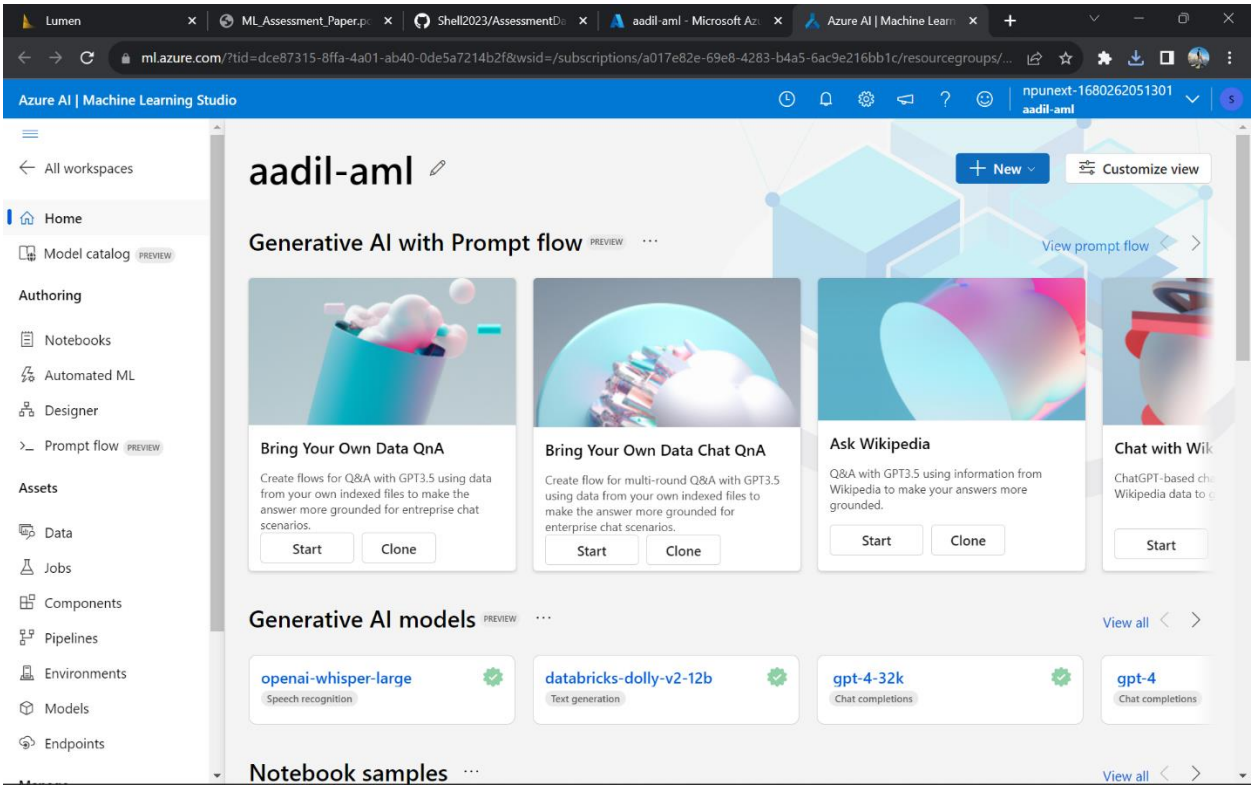
Deployment in progress



Deployment Complete

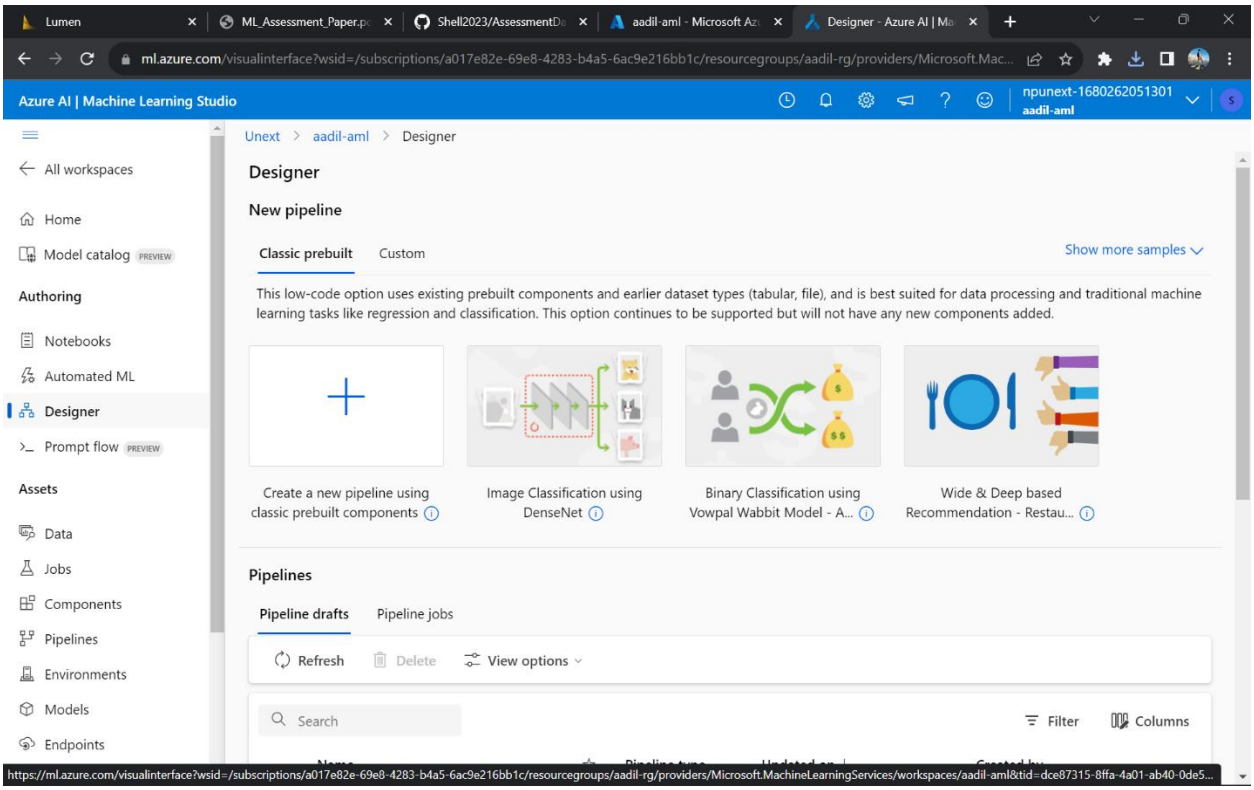


Resource

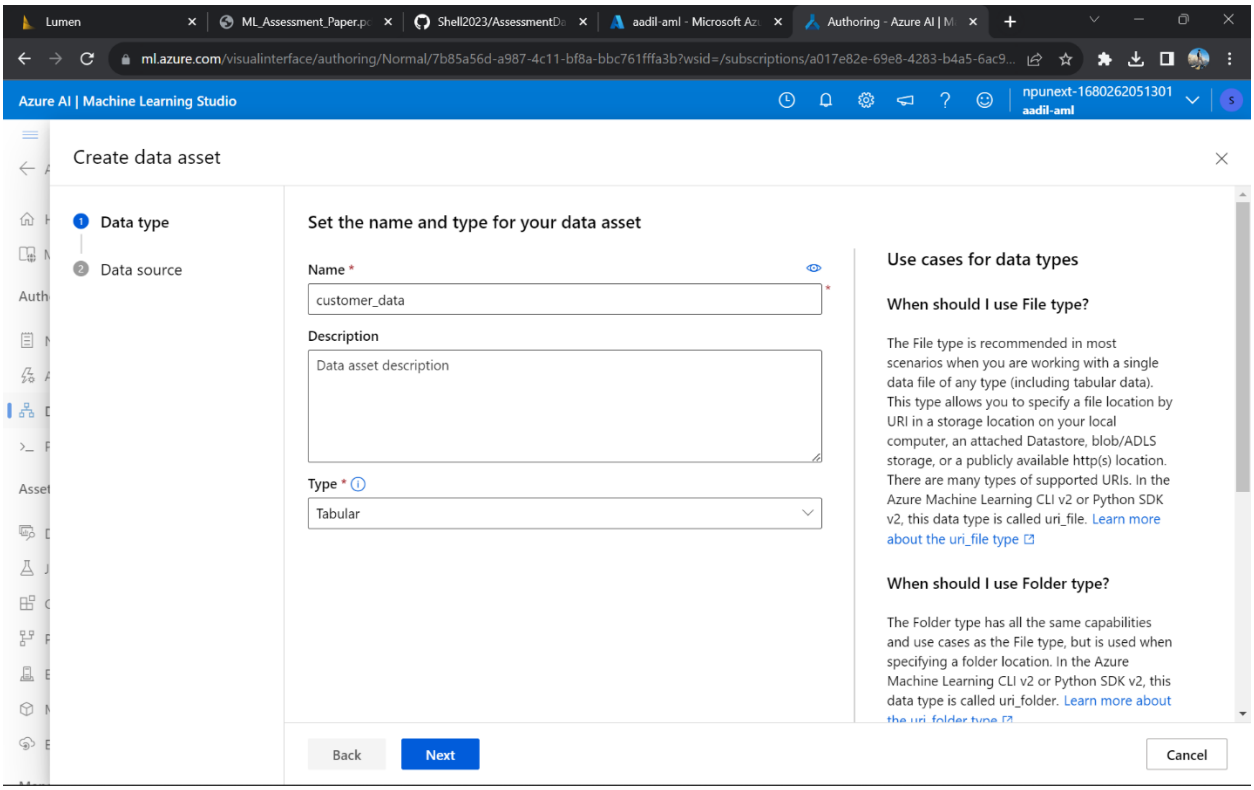


Azure ML Studio

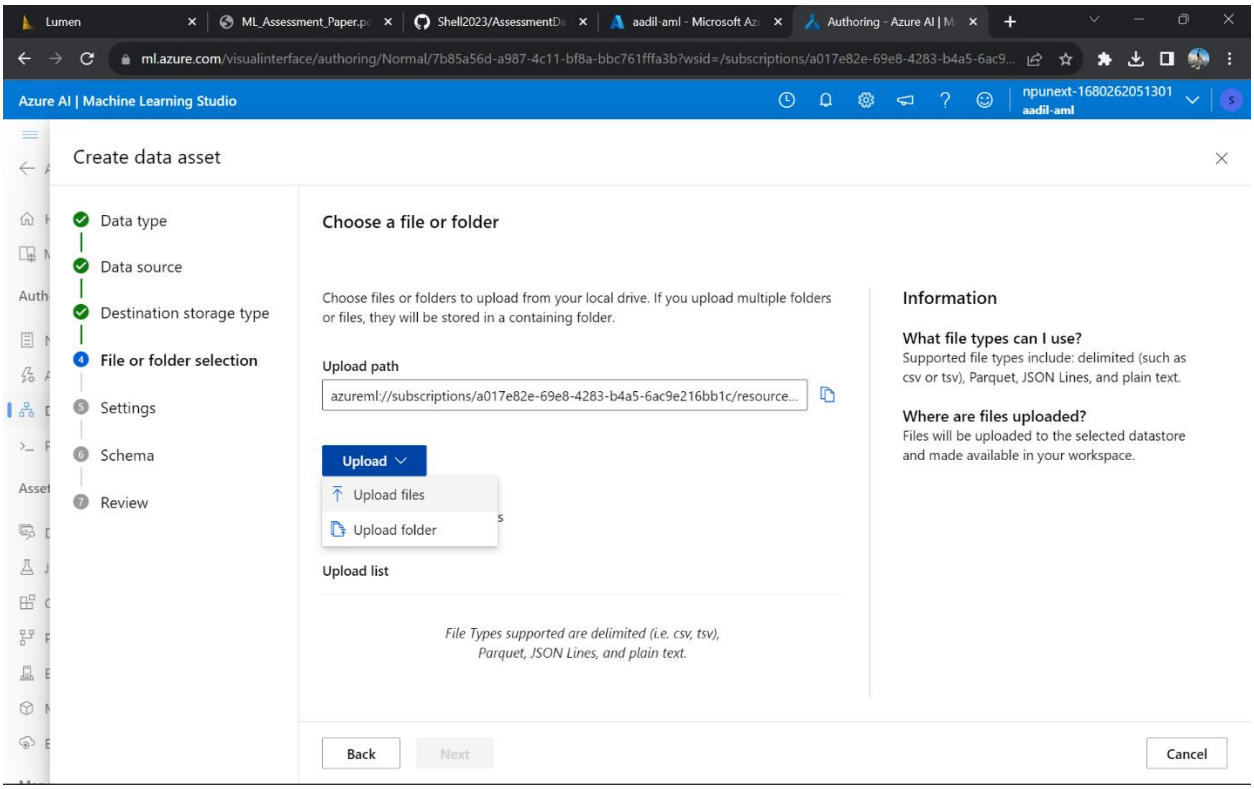
1. Data Preparation



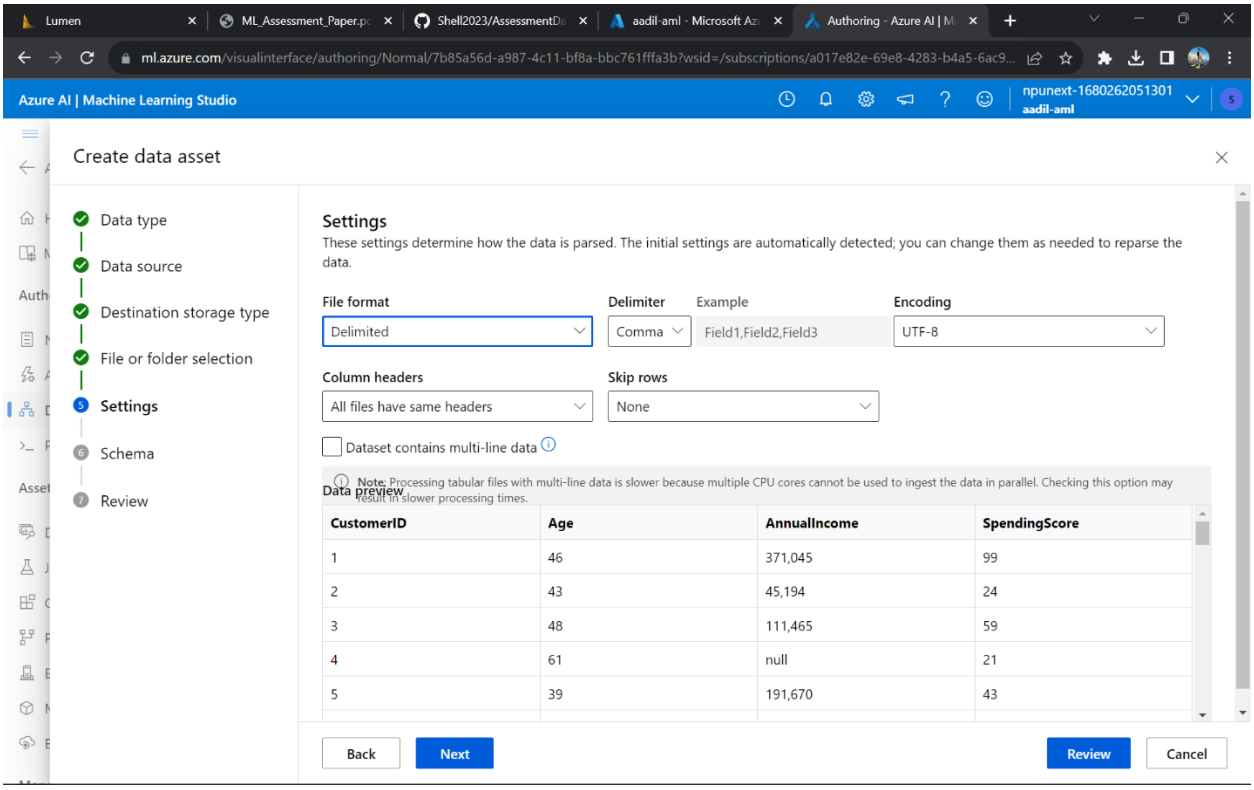
Azure ML Studio Designer



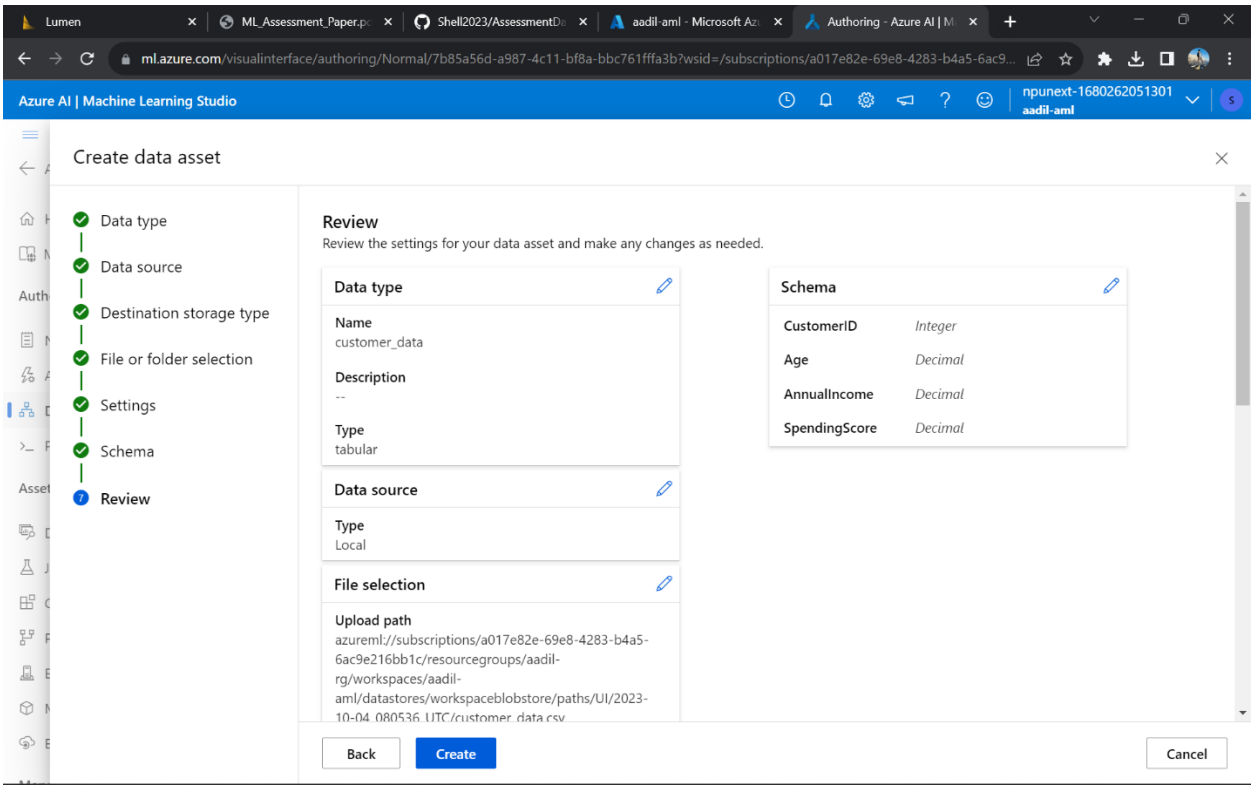
Creating Data-Asset



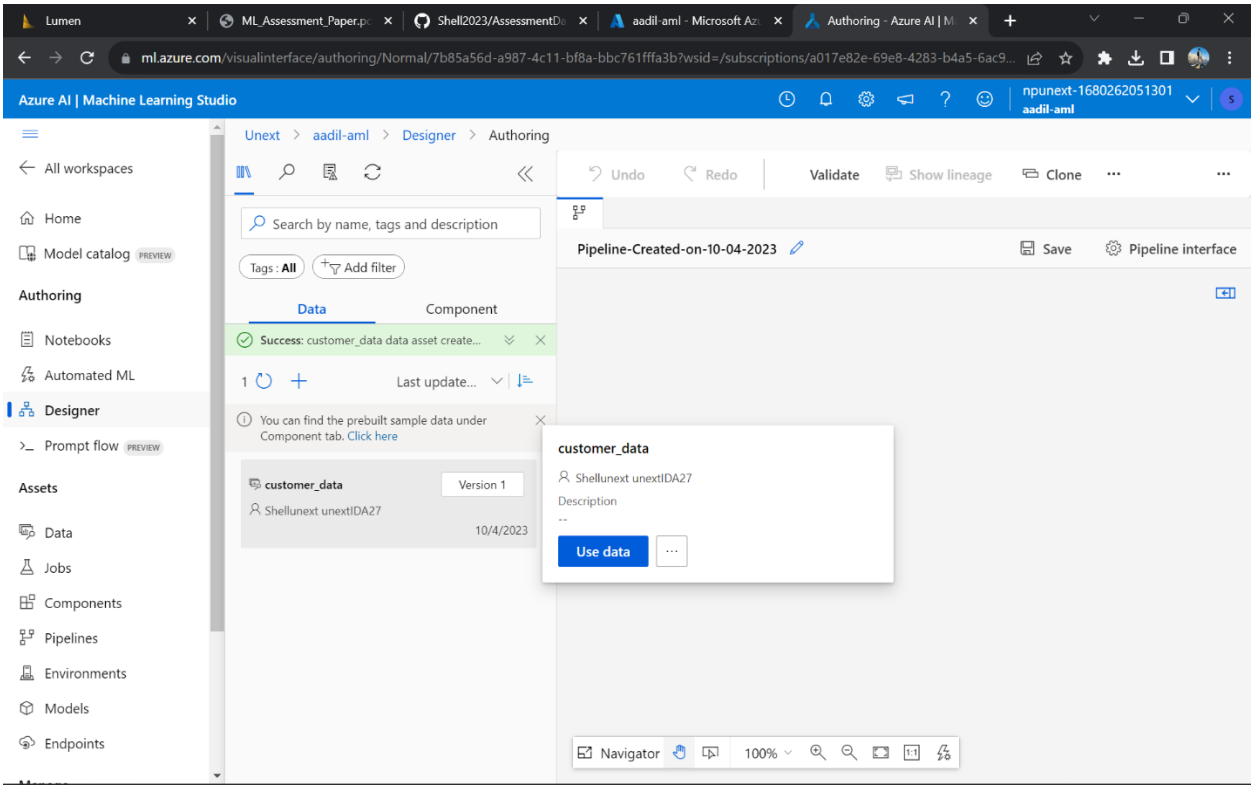
Uploading dataset



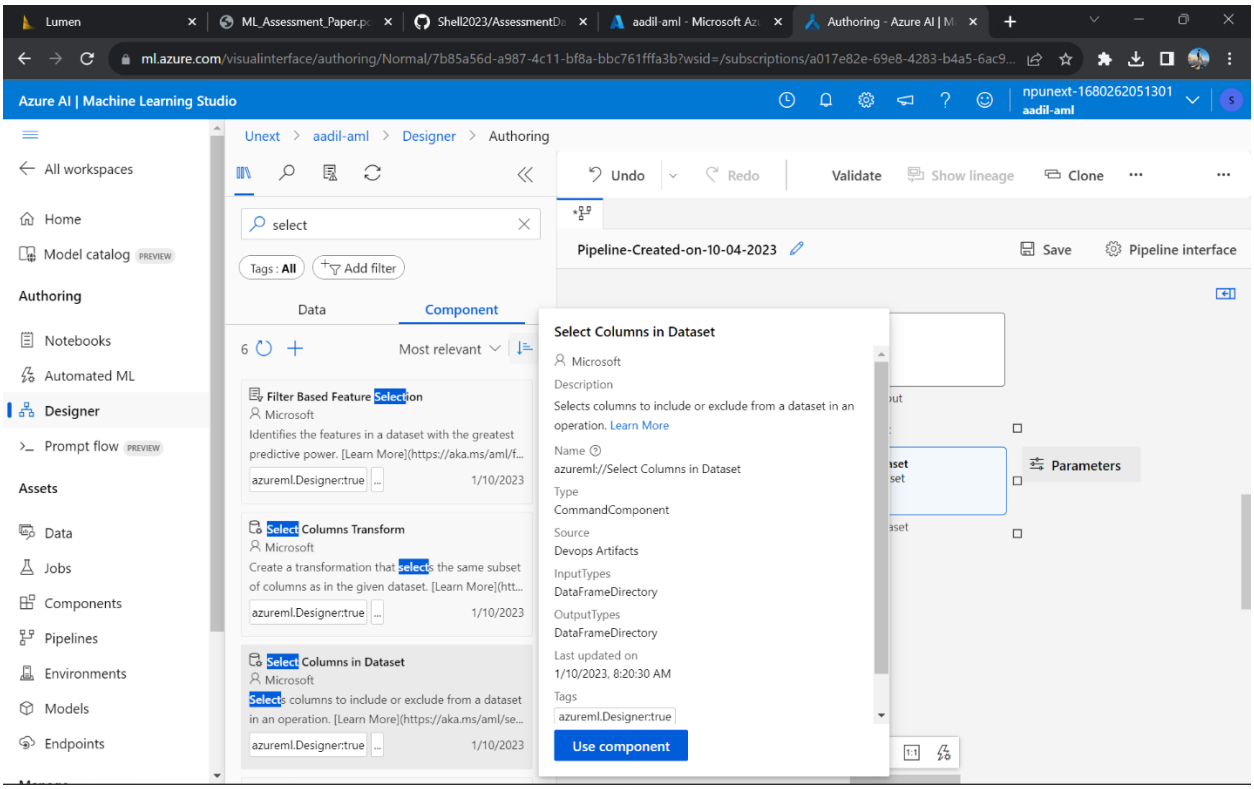
Validation



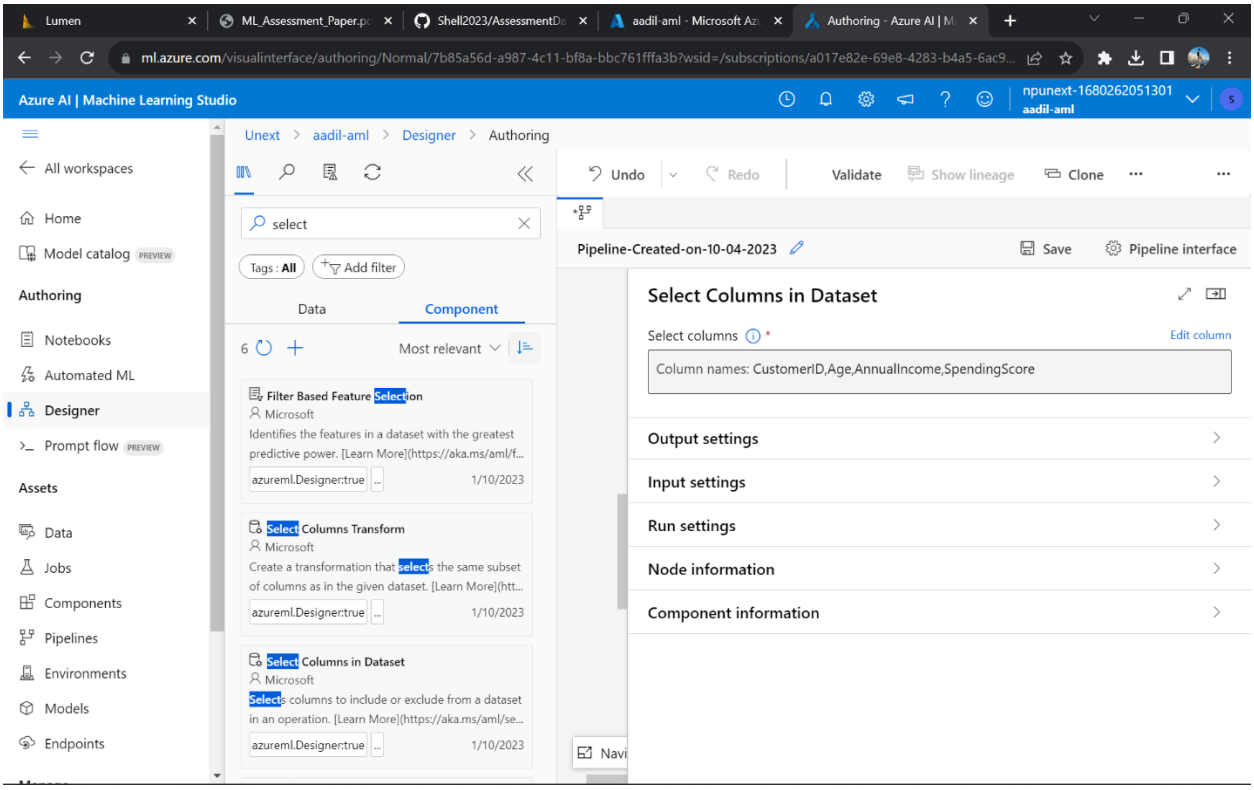
Create Data Asset



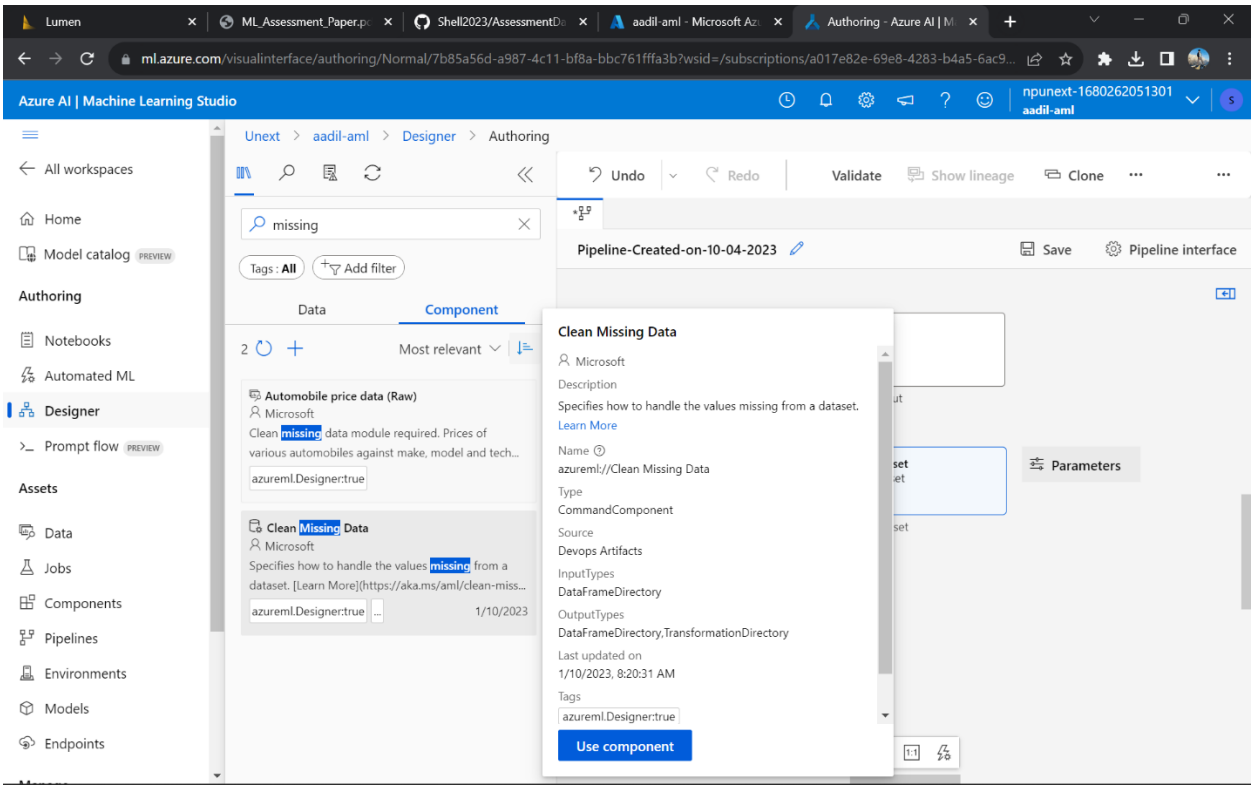
Select Dataset



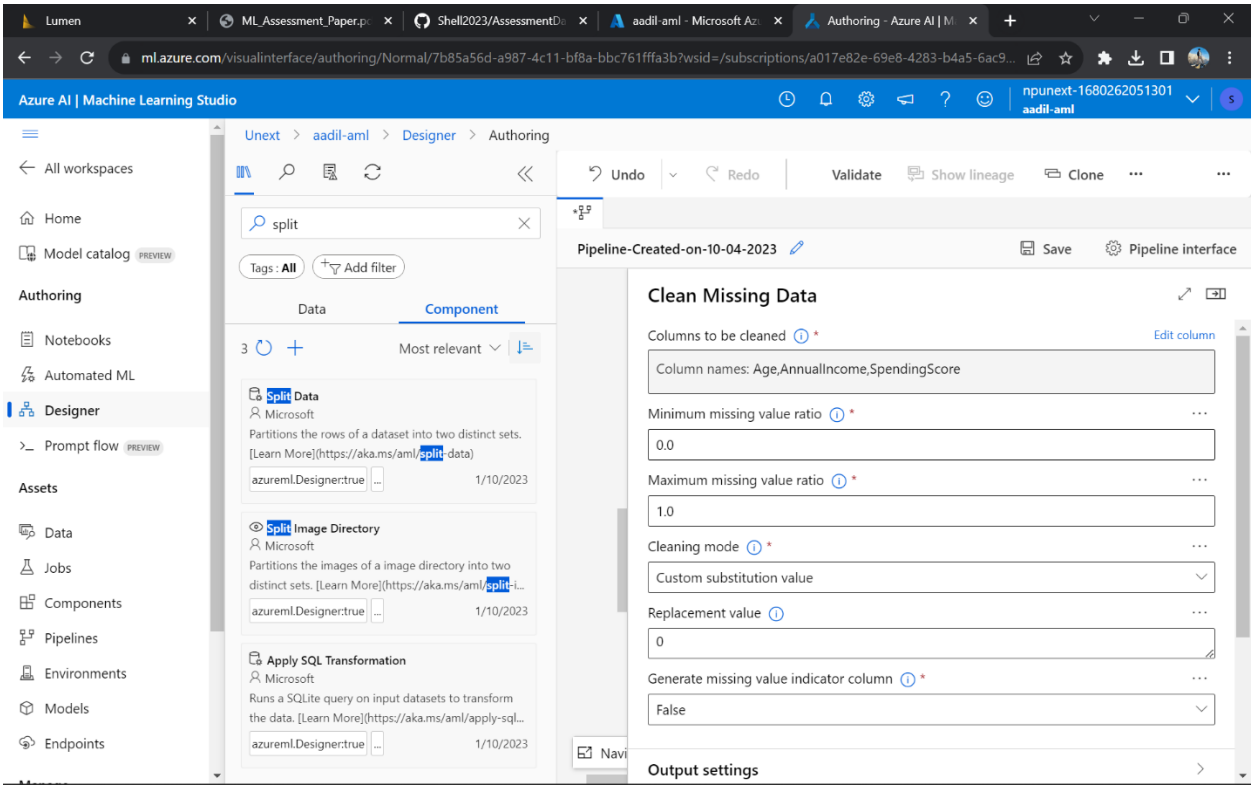
Select Column in Dataset



Select Columns

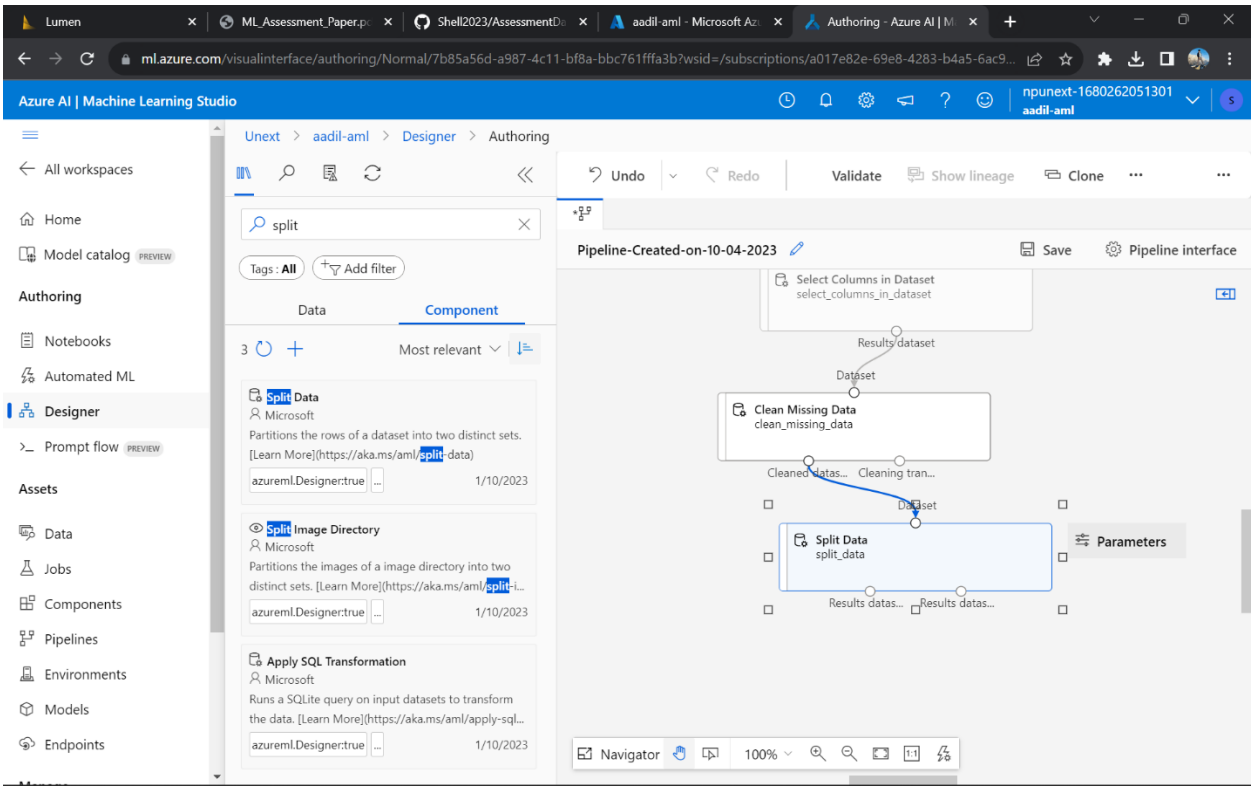


Clean Missing Data Component

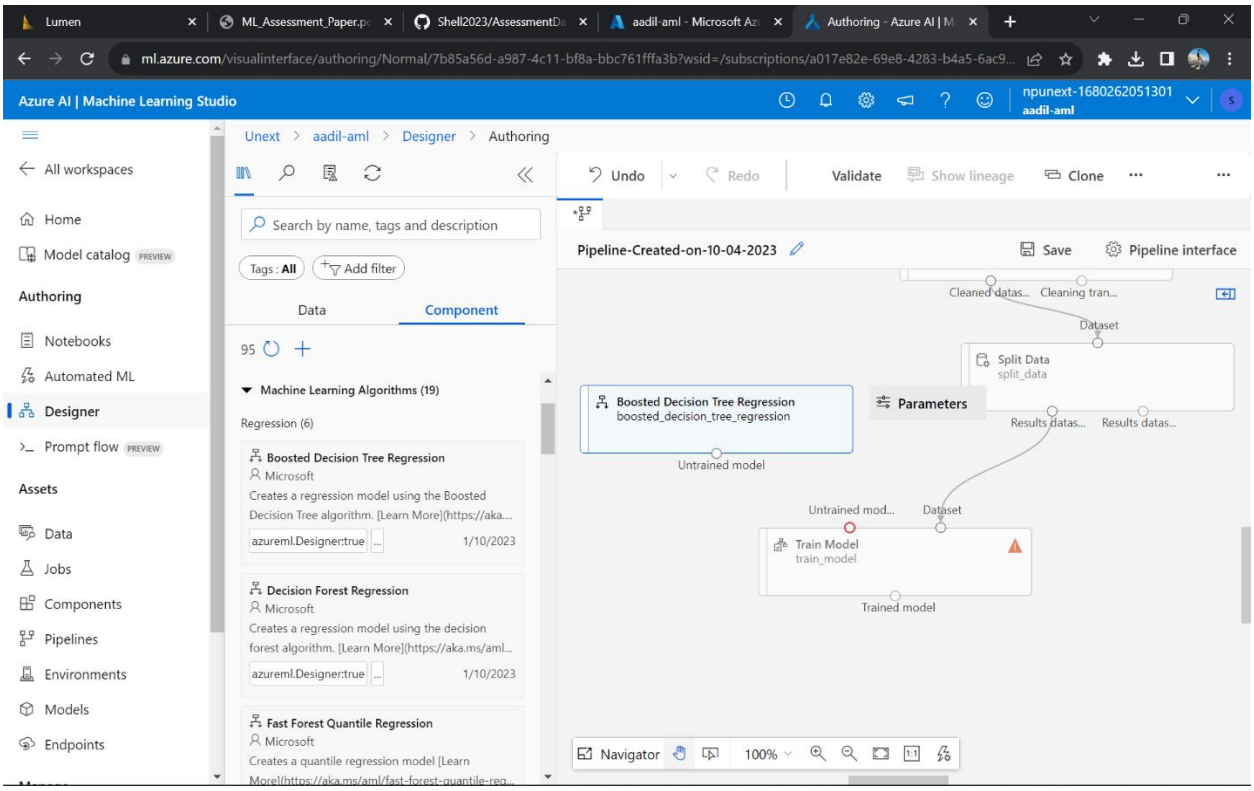


Configure Clean Data

2. Model Development

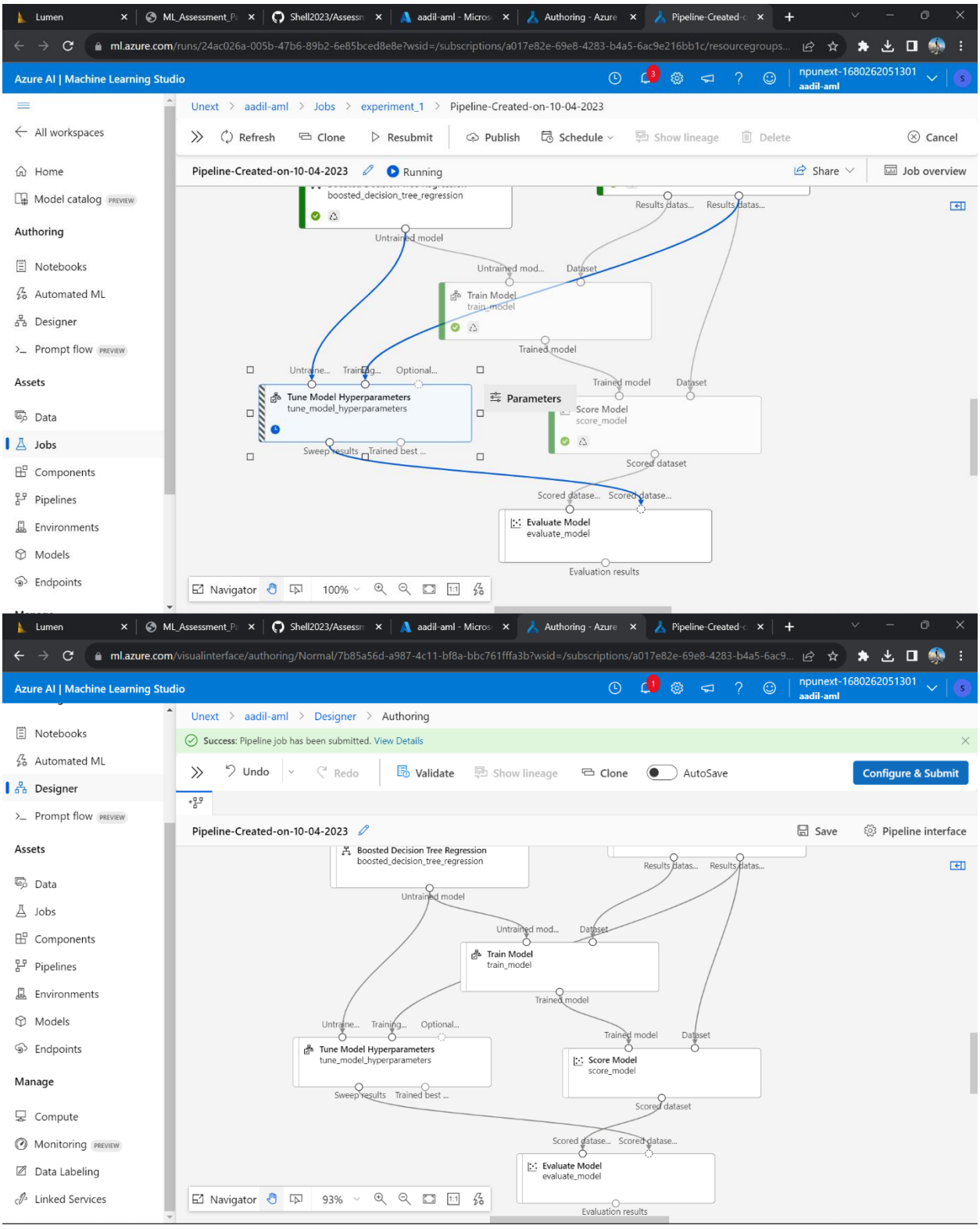


Split Data

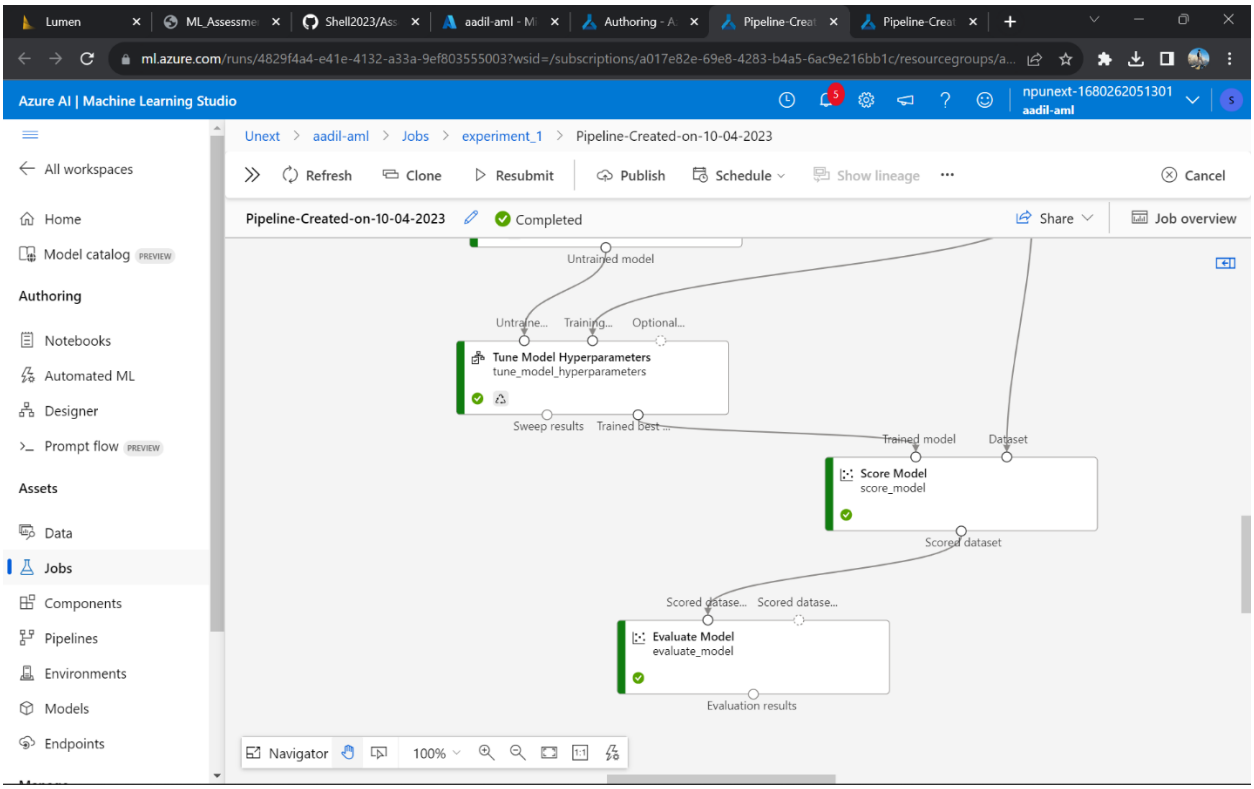


Machine Learning Algorithm

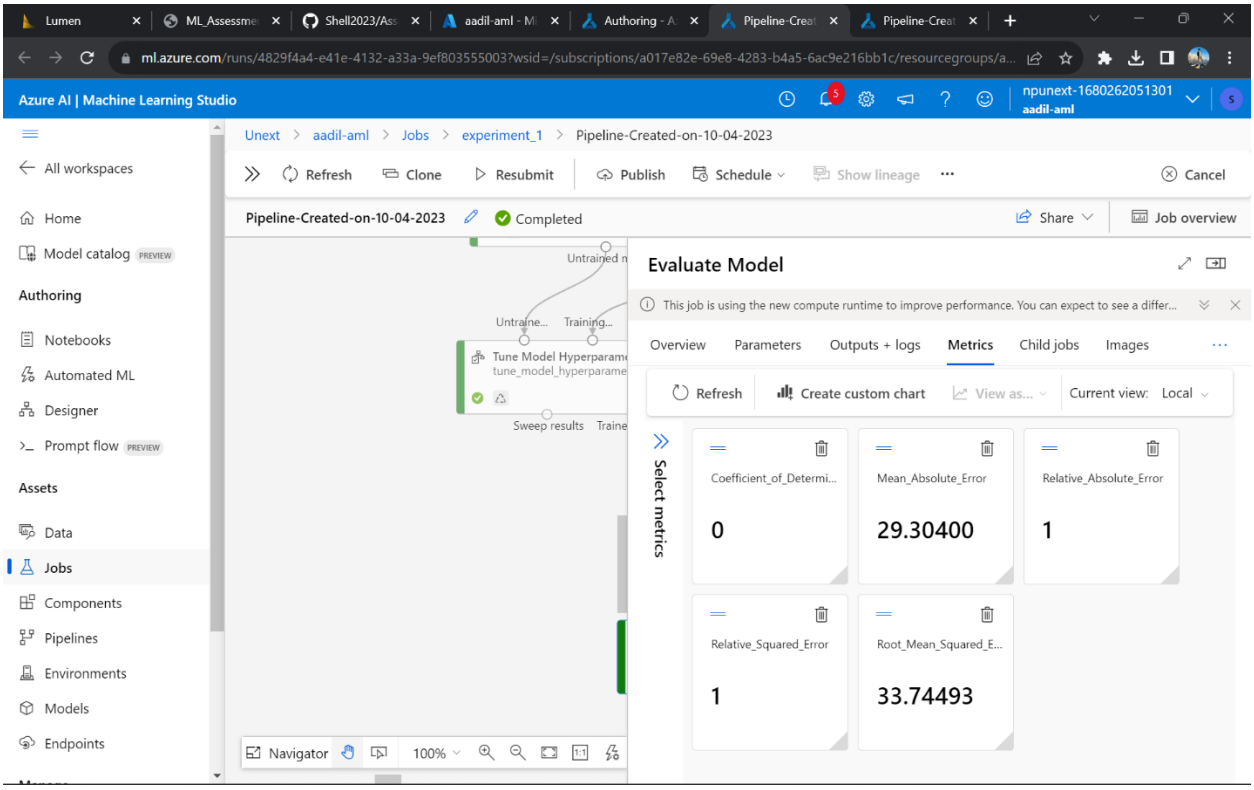
3. Hyperparameter Tuning



Train Model Hyperparameter, Score Model, Evaluate Model



Pipeline Ran Successfully



Evaluate Model Metrics

HANDS-ON COMPLETED

ASSESSMENT → QUESTION-ANSWERS

What are the key steps involved in preparing the dataset for training a machine learning model using Azure Machine Learning? Briefly explain each step.

Ans:

Preparing a dataset for training a machine learning model using Azure Machine Learning involves several key steps:

1. Data Collection and Ingestion:

- This initial step involves gathering the data from various sources, such as databases, files, or external APIs.

- In Azure Machine Learning, you can use Azure Data Factory or Azure Data Lake Storage to ingest and store your data.

2. Data Exploration:

- Before proceeding, it's essential to understand your data. This includes examining the data's structure, distribution, and quality.

- Use tools like Azure Databricks, Jupyter notebooks, or Azure Machine Learning Studio for data exploration.

3. Data Cleaning and Preprocessing:

- Clean the data by handling missing values, outliers, and duplicates.

- Perform preprocessing tasks like feature scaling, one-hot encoding, or text tokenization as needed.

4. Feature Engineering:

- Create new features or transform existing ones to improve the model's performance. This might involve domain-specific knowledge.

- Feature selection can also be part of this step to reduce dimensionality.

5. Data Splitting:

- Split the dataset into training, validation, and test sets. This helps evaluate the model's performance accurately.

- Azure Machine Learning provides built-in tools for data splitting and cross-validation.

6. Data Storage:

- Store the prepared dataset in a versioned and organized manner, typically in Azure Blob Storage or Azure Data Lake.

- This ensures traceability and reproducibility throughout the machine learning workflow.

7. Data Pipeline Creation:

- Build a data pipeline using Azure Machine Learning Dataflows or other orchestration tools to automate data preparation steps.

8. Data Validation and Monitoring:

- Continuously monitor the quality of your dataset, set up alerts for data drift, and retrain models when necessary.

1. Why is it important to split the dataset into training and testing sets when developing a machine learning model? How does this help in model evaluation?

Ans

Splitting the dataset into training and testing sets is a fundamental practice in machine learning for several important reasons:

1. Model Evaluation and Validation:

- By splitting the dataset, you can evaluate your machine learning model's performance on data it has never seen during training.

- This helps assess how well your model generalizes to unseen or new data, which is a crucial measure of its effectiveness.

2. Avoiding Overfitting:

- When a machine learning model is trained on a single dataset without a separate testing set, it can "overfit" to the training data.

- Overfitting means the model has learned to fit the training data's noise and anomalies rather than the underlying patterns. As a result, it performs poorly on new data.

- The testing set serves as an independent validation to detect overfitting. If the model performs well on the training set but poorly on the testing set, it's a sign of overfitting.

3. Hyperparameter Tuning:

- Splitting the dataset allows you to tune hyperparameters (e.g., learning rate, regularization strength) without using the testing data for this purpose.

- You can adjust hyperparameters on the training set and use the testing set to validate the chosen hyperparameter values.

4. Model Selection:

- When you have multiple candidate models, splitting the data helps you compare their performance on the same testing set.

- This allows you to select the best-performing model for your specific problem.

5. Bias and Variance Analysis:

- By evaluating a model on both training and testing sets, you can gain insights into its bias and variance.

- High training accuracy but low testing accuracy indicates high variance (overfitting), while low training and testing accuracy suggest high bias (underfitting).

6. Confidence in Results:

- Splitting the data and evaluating on a separate set provides a more accurate assessment of your model's true performance.

- It gives you confidence that the reported performance metrics are representative of how the model will perform in practice.

3. Describe a machine learning algorithm suitable for predicting customer purchasing behaviour in the given scenario. Explain why you chose this algorithm.

Ans

Predicting customer purchasing behavior is a common use case in marketing and e-commerce. One suitable machine learning algorithm for this scenario is the Random Forest – Boosted algorithm.

1. Ensemble Method: Random Forest is an ensemble learning method that combines multiple decision trees to make predictions. Each decision tree is trained on a subset of the data, and their predictions are averaged or voted upon to produce the final result. This ensemble approach often results in more robust and accurate predictions.

2. Handling Complex Relationships: Customer purchasing behavior can be influenced by a multitude of factors, some of which may have complex and non-linear relationships. Random Forests can capture these complex relationships as they can model both linear and non-linear patterns in the data.

3. Feature Importance: Random Forests provide a measure of feature importance, which is valuable in understanding which factors are most influential in predicting customer behavior. This information can be used for business insights and decision-making.

4. Reduced Overfitting: Random Forests are less prone to overfitting compared to single decision trees. By aggregating the results of multiple trees, they reduce the variance and improve generalization to new, unseen data.

5. Robustness to Outliers: Random Forests are robust to outliers and noisy data, which can be common in real-world customer behavior datasets.

6. Scalability: Random Forests can handle large datasets efficiently, making them suitable for scenarios with a significant amount of customer data.

4. What is hyperparameter tuning, and why is it important in machine learning? Explain a technique used for hyperparameter tuning and its benefits.

Ans

Hyperparameter tuning is the process of optimizing the hyperparameters of a machine learning algorithm to achieve the best possible model performance. Hyperparameters are parameters that are not learned from the data during training but are set before the training process begins. They control aspects of the learning process, such as the model's complexity, convergence speed, and generalization ability.

Hyperparameter tuning is essential in machine learning for several reasons:

1. Performance Improvement: Properly tuned hyperparameters can significantly improve a model's performance. They can make the difference between a model that underfits (too simple) or overfits (too complex) the data and one that generalizes well to new, unseen data.

2. Generalization: Optimized hyperparameters help the model generalize better to new data, reducing the risk of overfitting or underfitting. This leads to more reliable and robust models.

3. Efficiency: Hyperparameter tuning can lead to faster convergence during training, making the model learn more efficiently. This is particularly important when dealing with large datasets or complex models.

4. Model Selection: In cases where multiple machine learning algorithms are available, tuning hyperparameters can help select the best algorithm for a specific problem by comparing their performance under optimized settings.