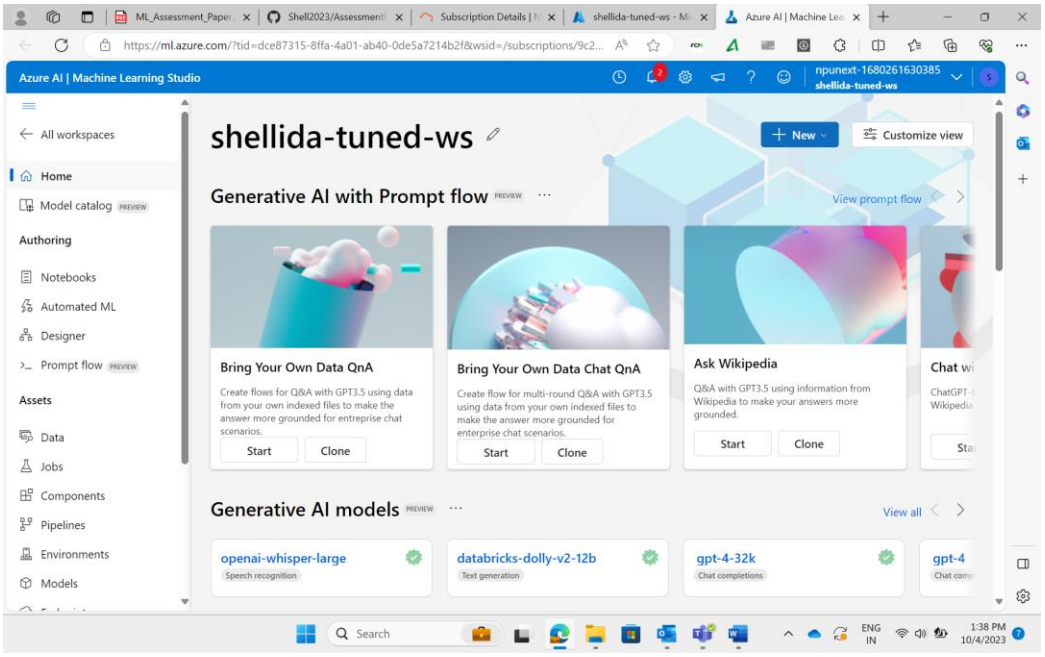
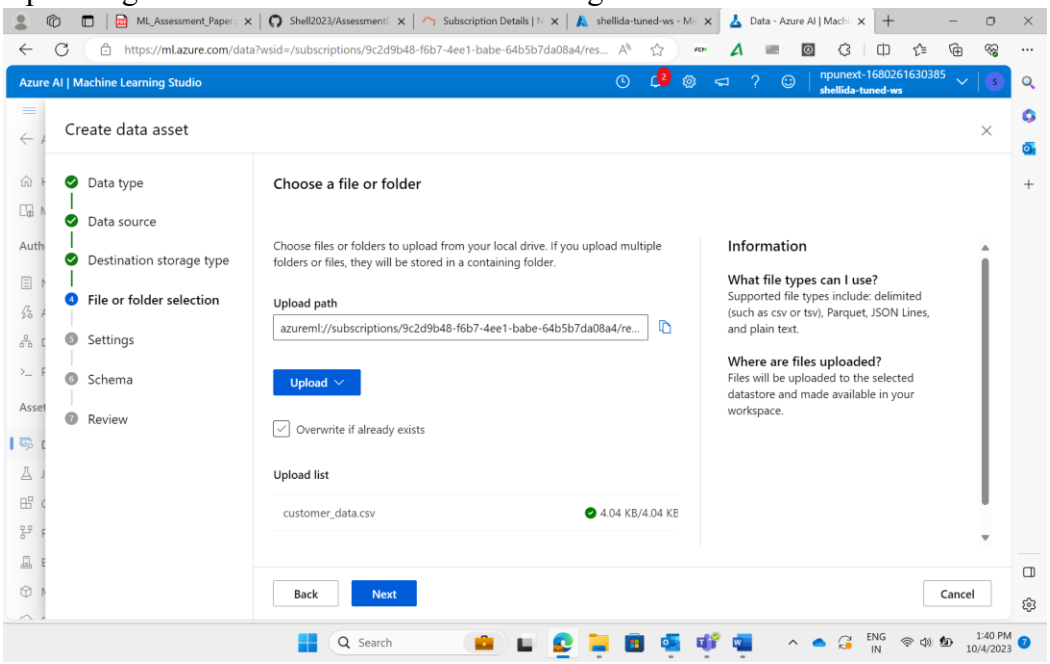
AZURE MACHINE LEARNING ASSESSMENT

- ANISHA GUPTA
EMP ID: 655517

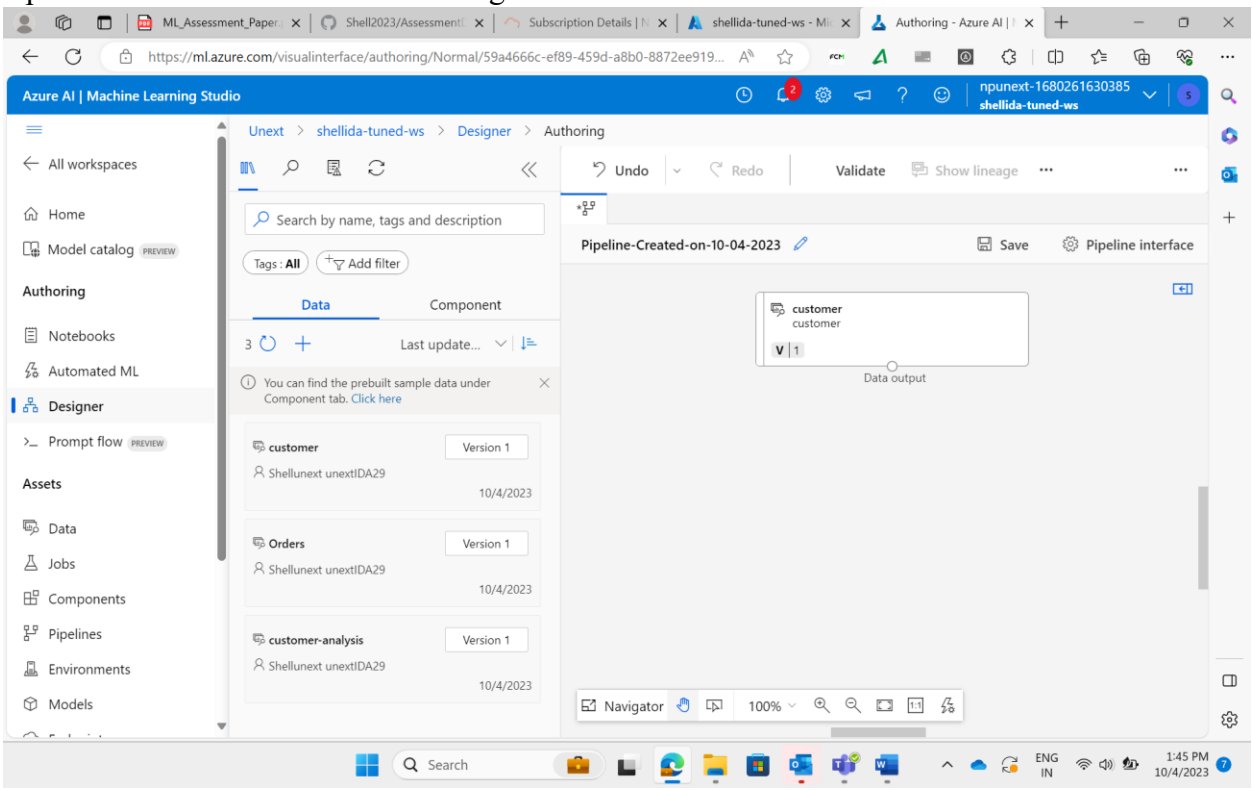
Opened Azure ML Studio



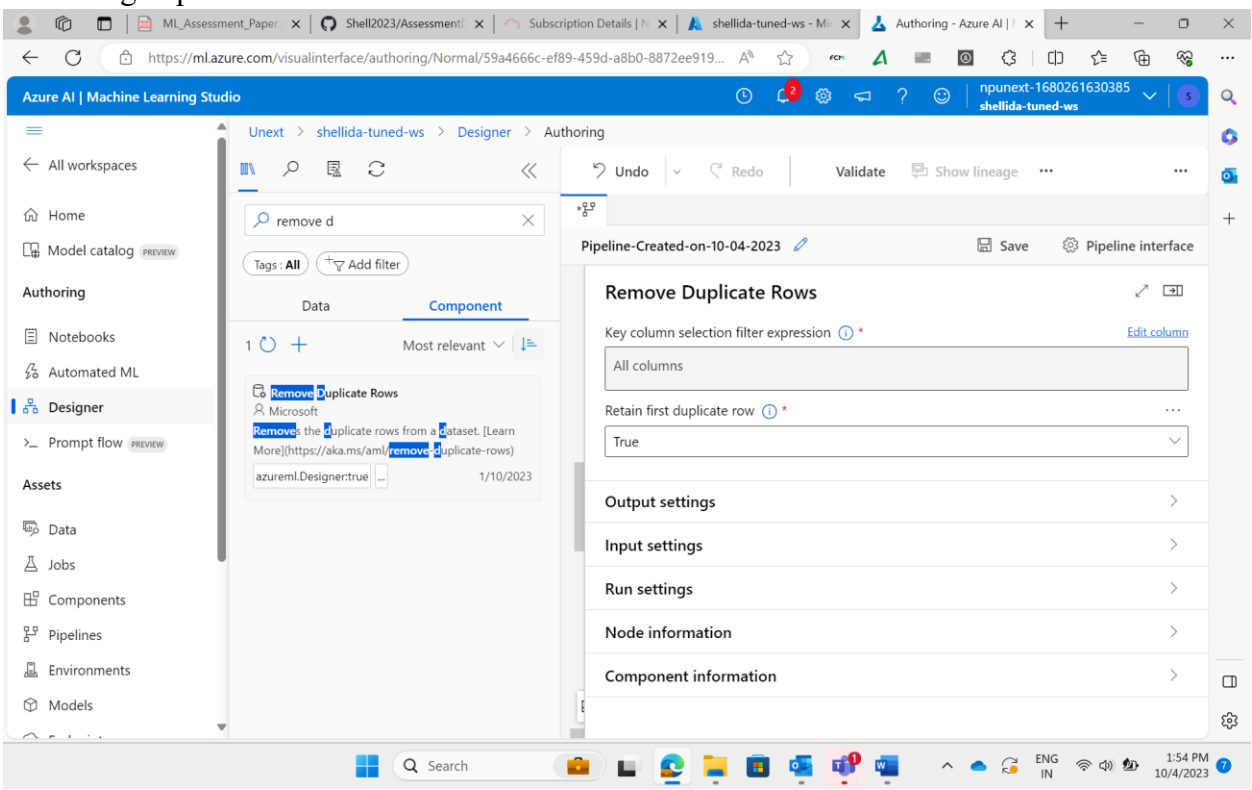
Uploading customer data to Azure Blob Storage



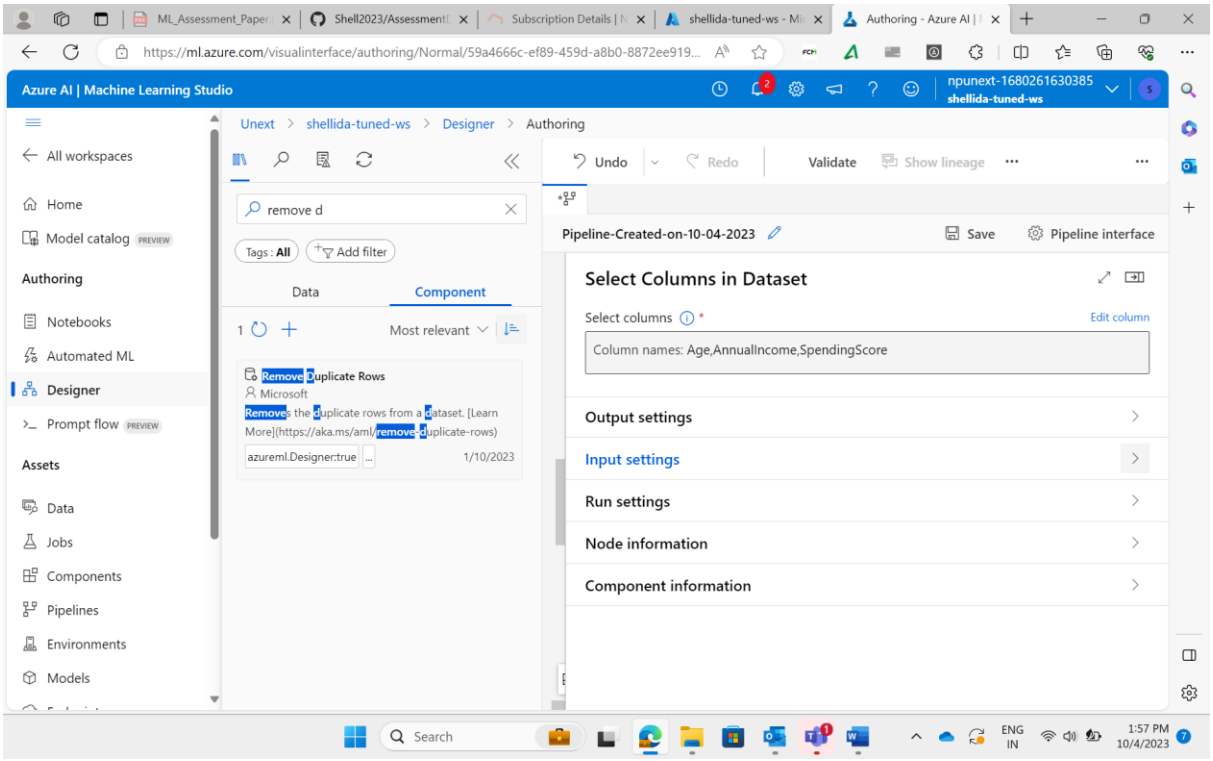
Uploaded data from Azure Blob Storage:



Removing duplicate rows:

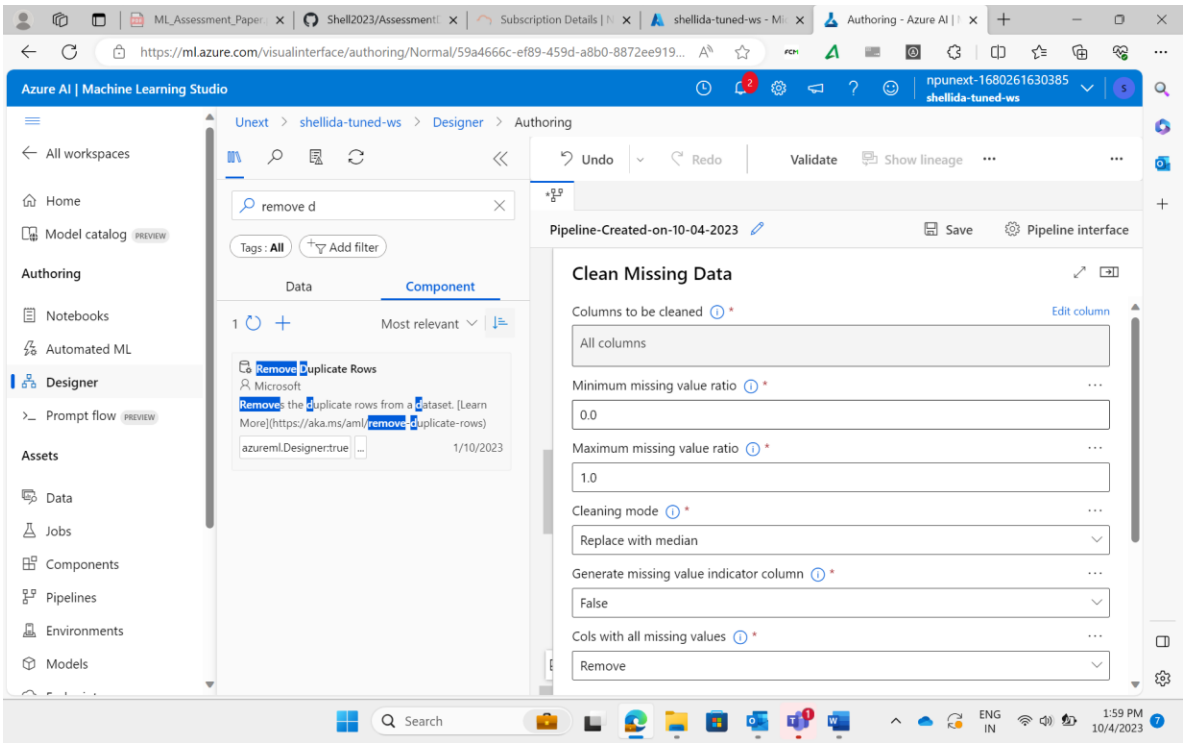


Selected all columns from the dataset:

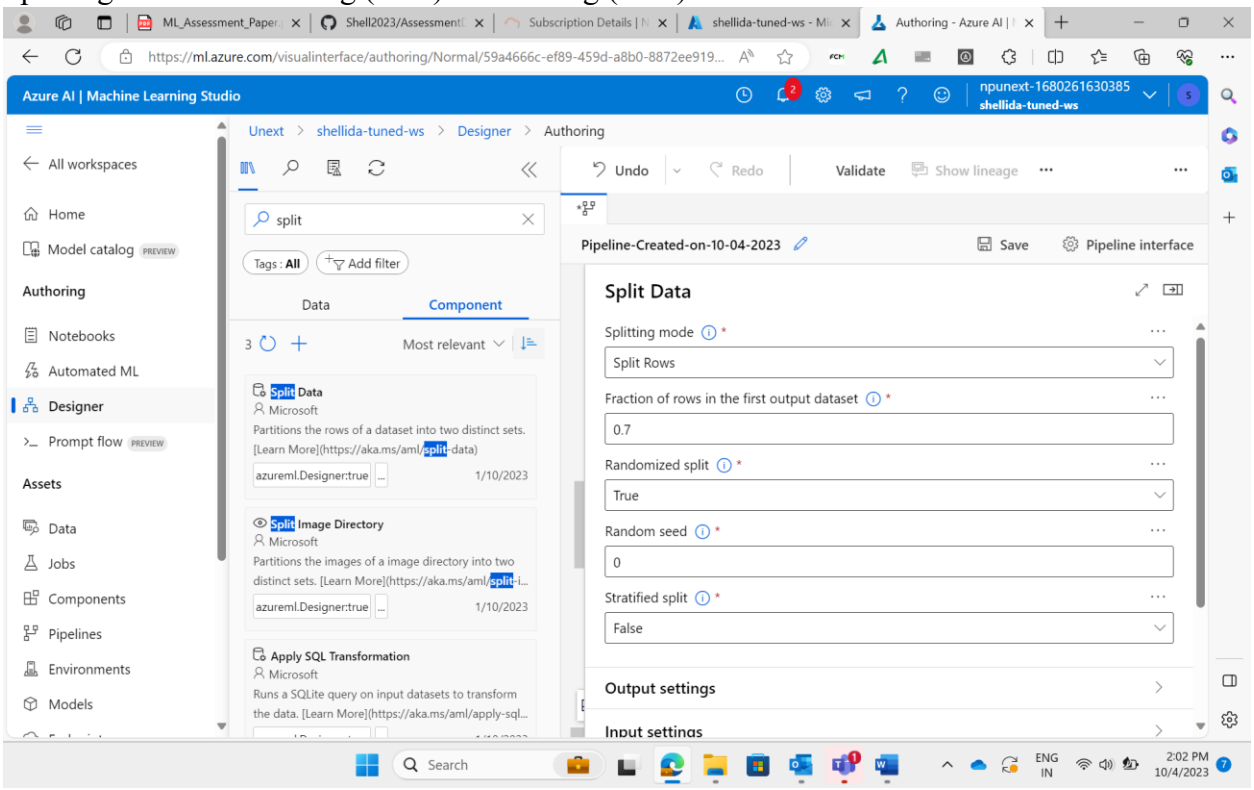


Removed Id, and kept the rest of the columns

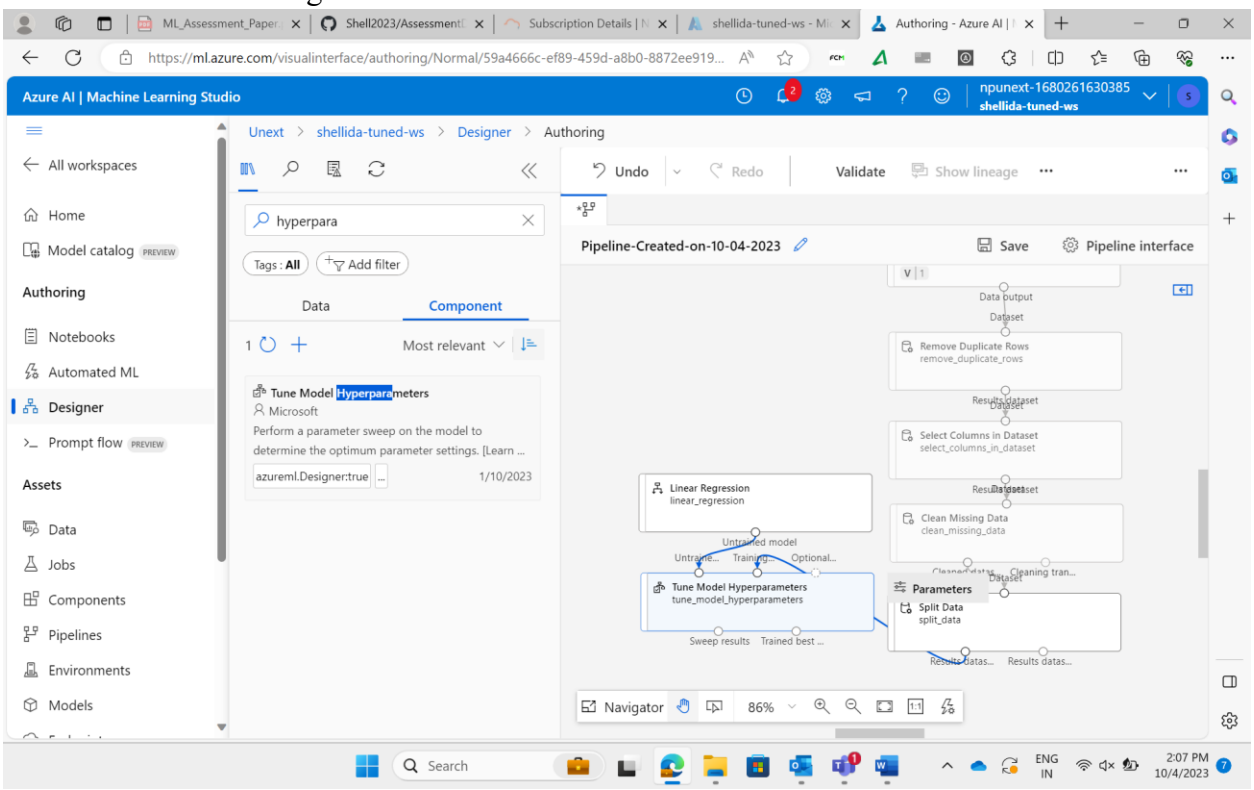
Cleaning the missing data:



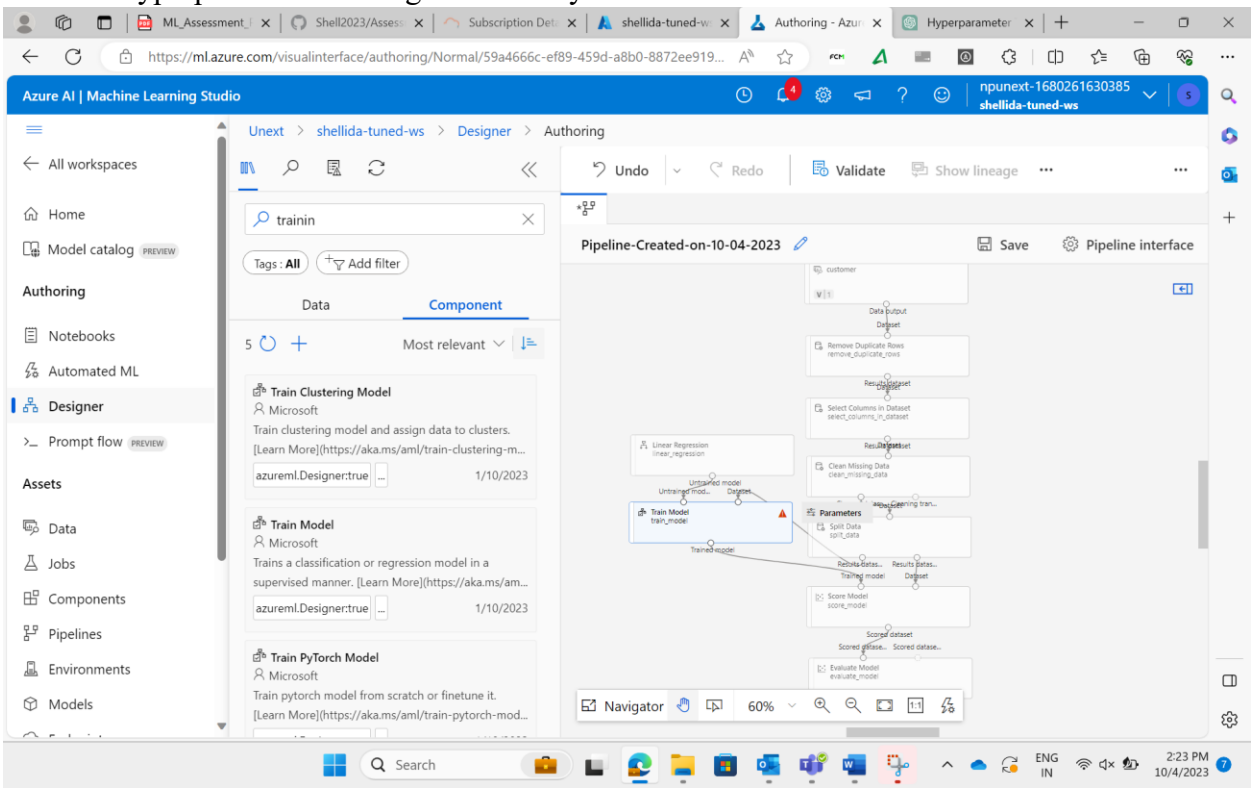
Splitting Data into Training (70%) and Testing (30%)

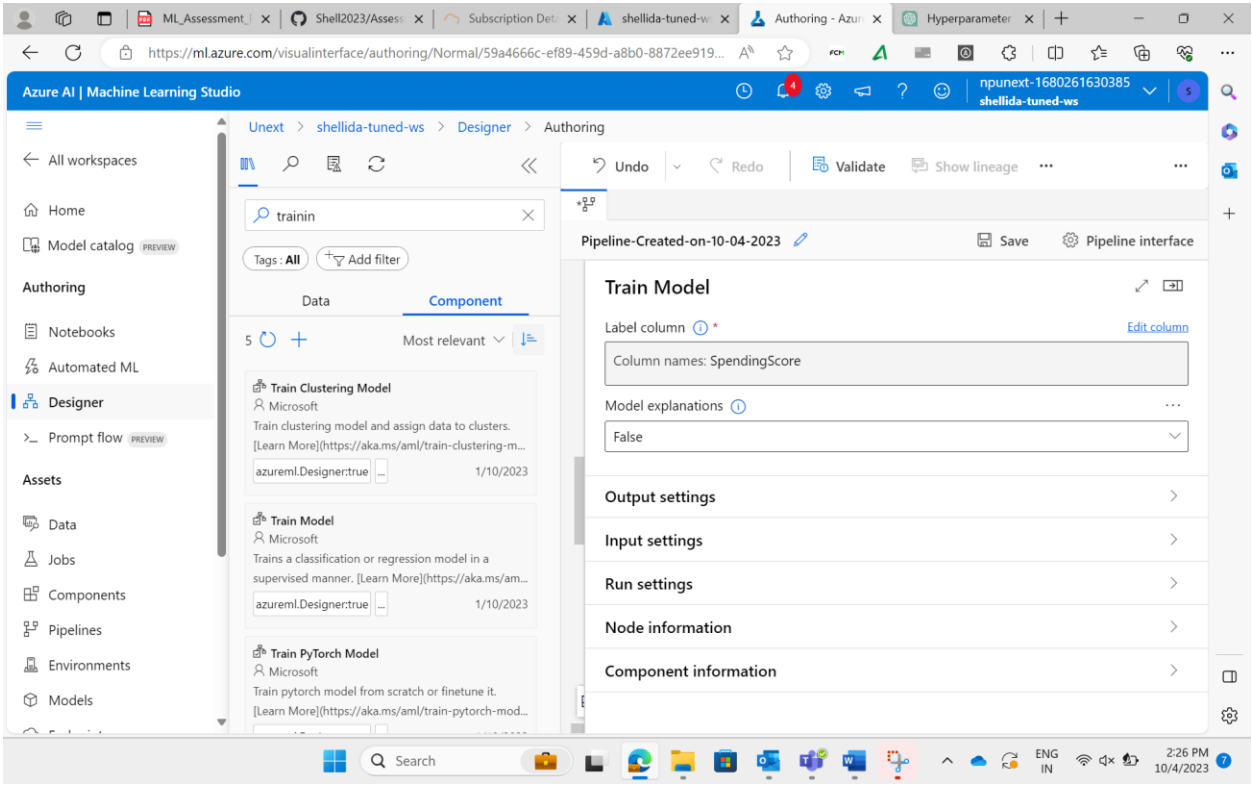


Train Model: Linear Regression

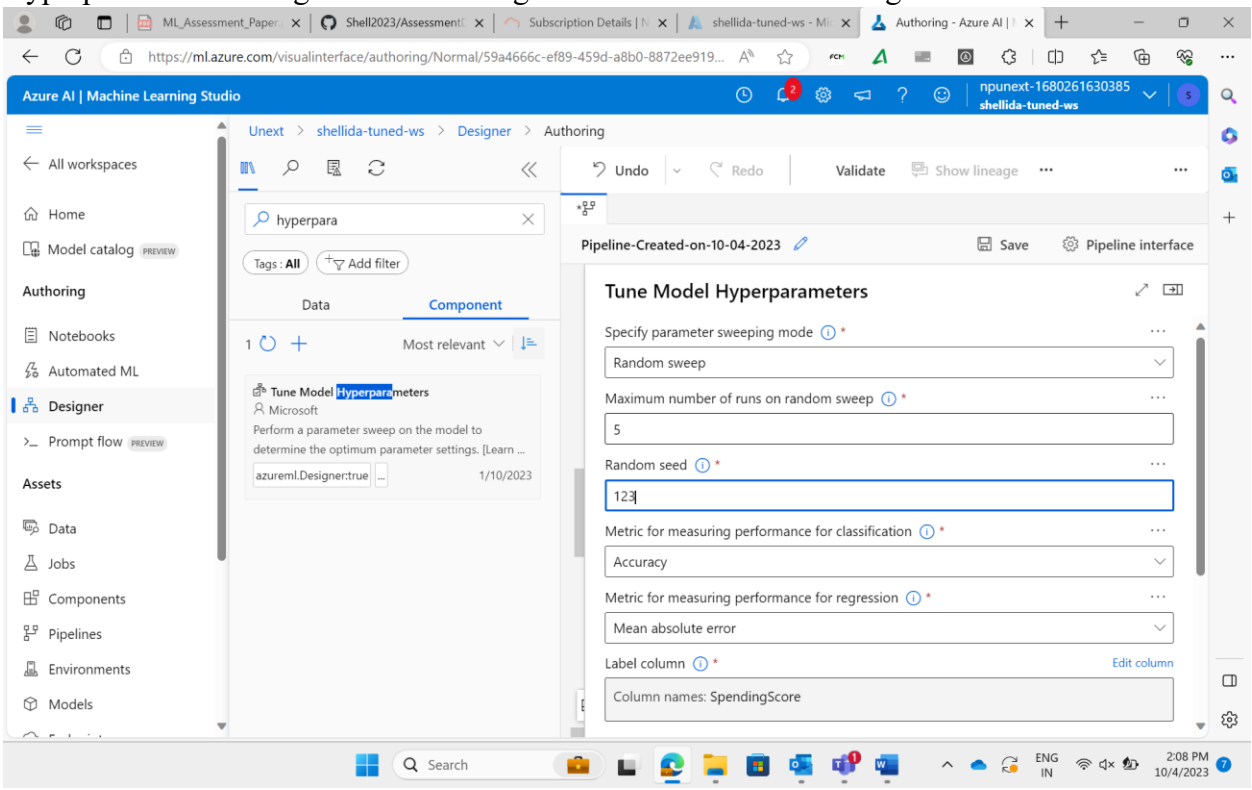


Without Hyperparameter Tuning: We directly train data

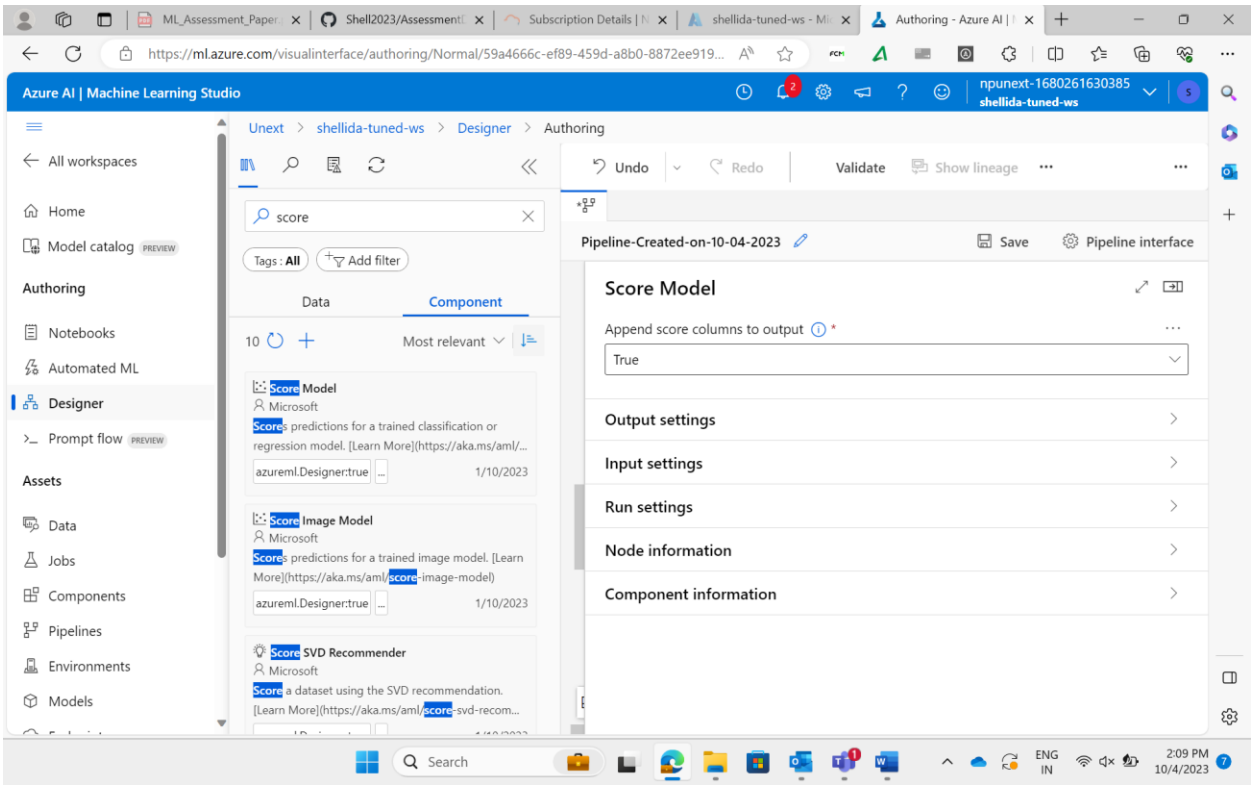




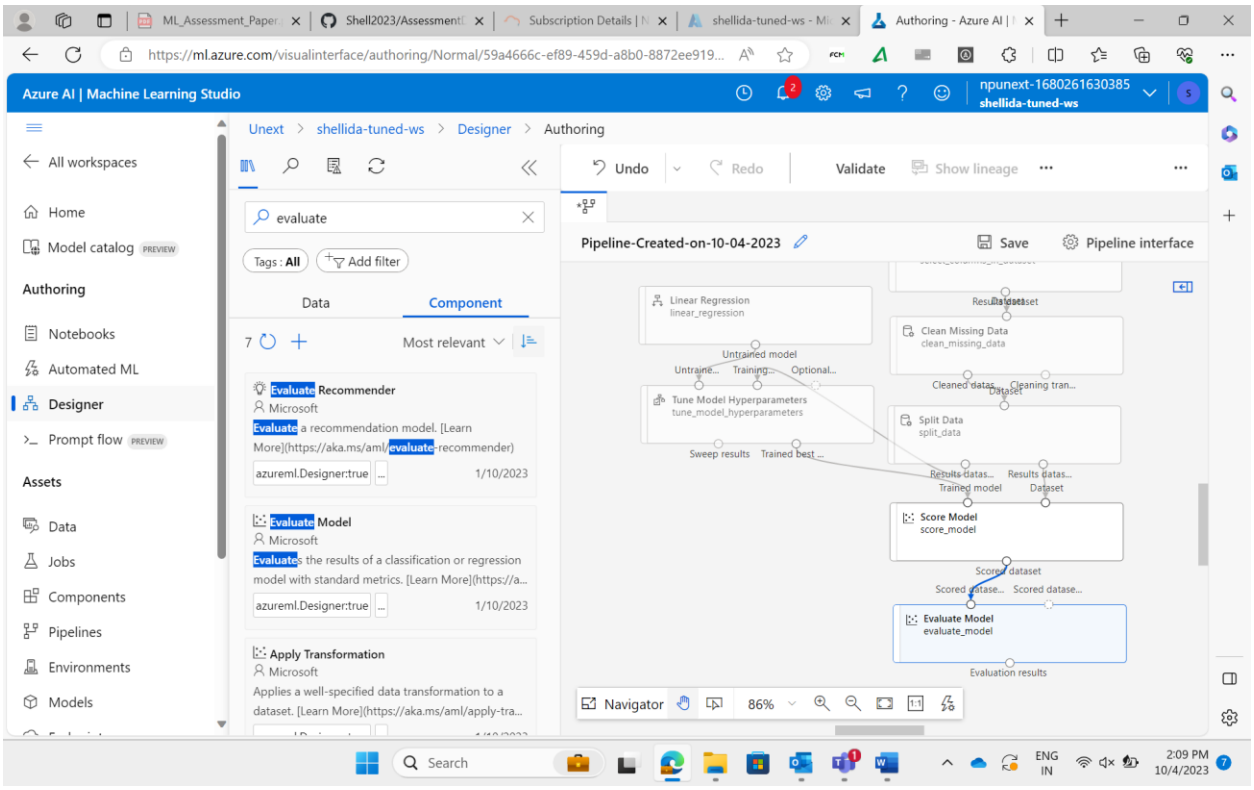
Hyperparameter Tuning of Model: Using Mean Absolute Error for regression



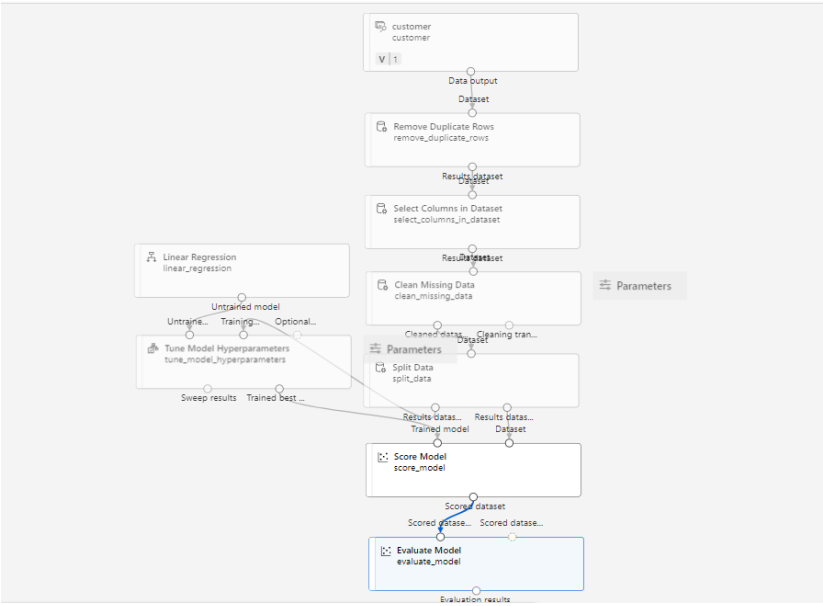
Score Model:



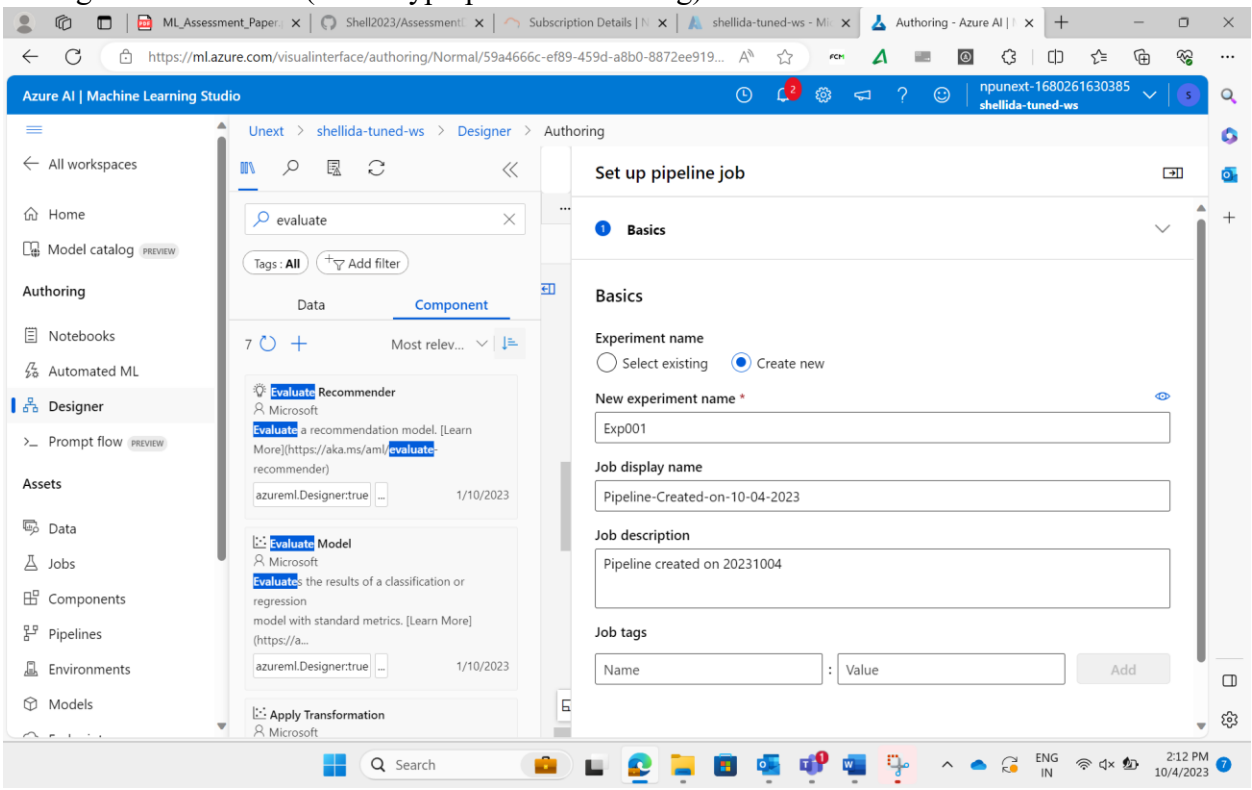
Evaluate Model:



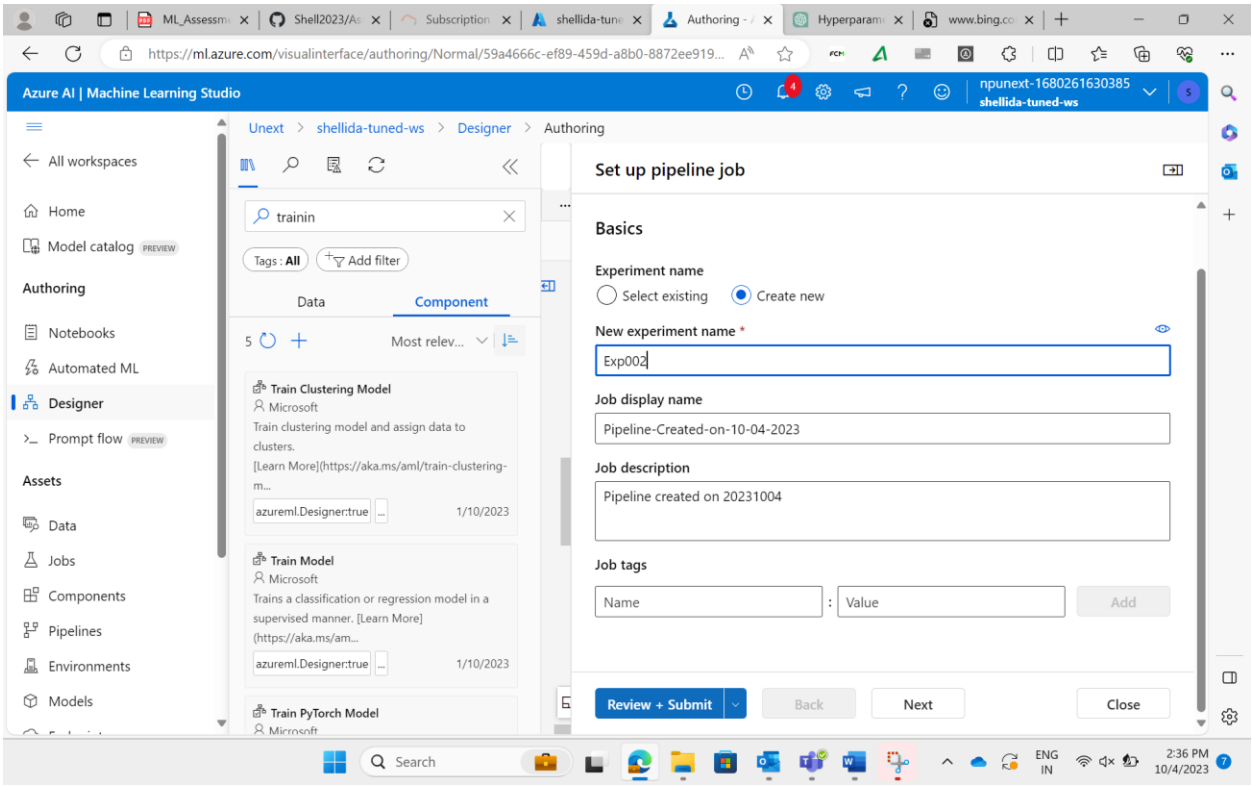
The entire pipeline:



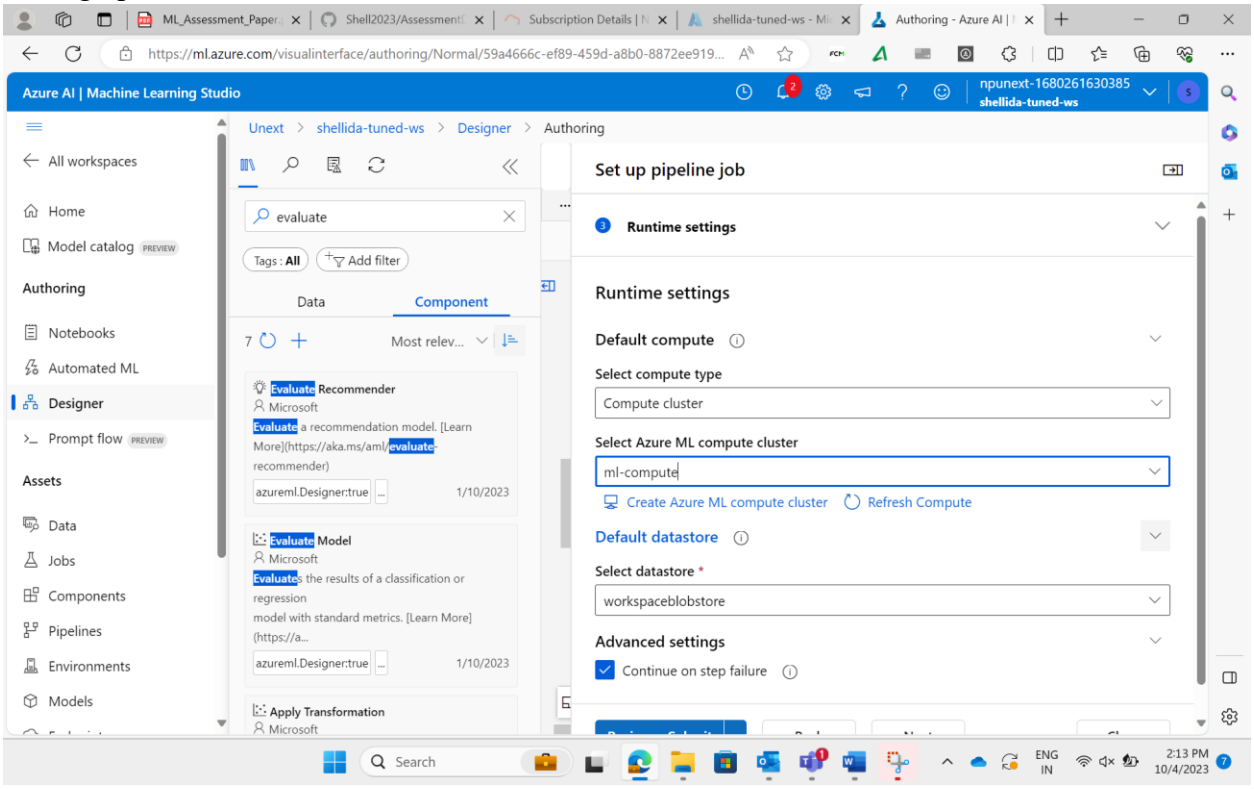
Configure and Submit: (With Hyperparameter Tuning)

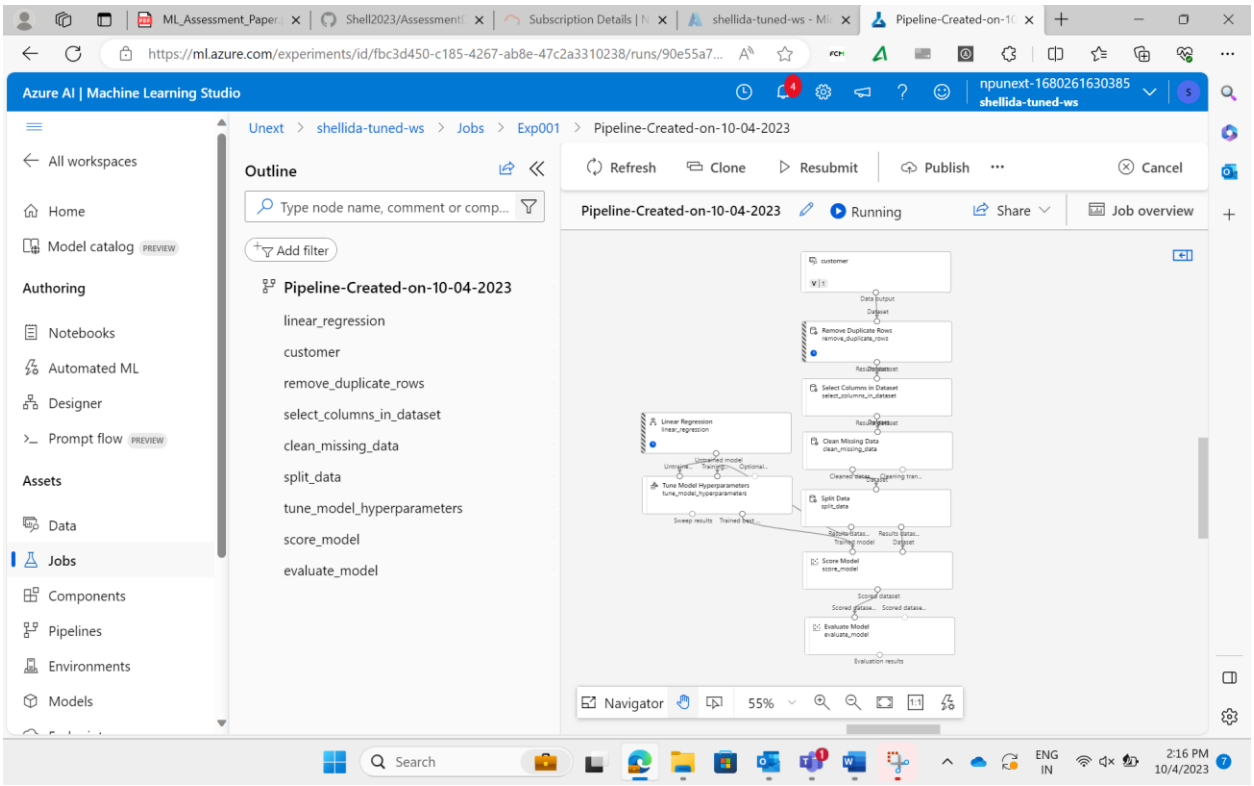


Without Hyperparameter Tuning



Setting up the runtime environment:





Similarly Exp002 is also in running state

Assessment Questions:

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

Data Ingestion:
Ingest or import the data into Azure ML

Data Exploration and Analysis:
Perform exploratory data analysis (EDA) to gain a better understanding of your dataset. Explore summary statistics, data distributions, missing values, and potential outliers. Visualization tools in Azure ML can help with this step.

Data Cleaning:
Handle missing values, duplicate records, and outliers. Impute missing values, remove duplicates, and apply appropriate techniques to address outliers. Data cleaning is essential to ensure the quality of the dataset.

Feature Engineering:
Create, transform, or select features (variables) that are relevant to your machine learning task. Feature engineering can involve techniques such as one-hot encoding, scaling, or creating new features from existing ones.

Data Splitting:
Split the dataset into multiple subsets for training, validation, and testing purposes. Common splits include a training set for model training, a validation set for hyperparameter tuning and model selection, and a test set for evaluating final model performance.

Data Preprocessing:
Apply preprocessing techniques such as normalization, standardization, or feature scaling to make the data suitable for machine learning algorithms. Preprocessing ensures that all features have the same scale and distribution.

Data Transformation:
Perform any necessary data transformations, including encoding categorical variables, handling time series data, or applying dimensionality reduction techniques like Principal Component Analysis (PCA).

Data Validation:

Validate the prepared dataset to ensure that it meets the requirements of your machine learning problem. Check for any data leakage or issues that might affect model performance.

Data Storage:
Store the cleaned and preprocessed dataset in Azure ML's data storage solutions for easy access during model training and deployment.

Data Feeding to Model:

Once your dataset is prepared, you can feed it into the machine learning model training process in Azure ML, using the appropriate model training algorithm and techniques. Evaluate the result from the model using the scores.

2. **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?**

Splitting the dataset into training and testing sets is a fundamental step in developing a machine learning model, and it serves several important purposes in the model development process. Here's why it's crucial and how it helps in model evaluation:

Assessing Generalization Performance: The primary goal of a machine learning model is to make accurate predictions on new, unseen data. By splitting the dataset into two distinct sets, you can evaluate how well your model generalizes to unseen data

Avoiding Overfitting: Overfitting occurs when a model learns to memorize the training data rather than capturing its underlying patterns. The testing set acts as a check against overfitting by evaluating the model's performance on data it has not seen during training.

Model Selection: Splitting the data allows you to compare different machine learning models or hyperparameter settings. This helps you choose the best-performing model or configuration.

Performance Evaluation: By having a separate testing set, you can calculate various performance metrics, such as accuracy, precision, recall, F1 score, or mean squared error, to quantitatively assess how well your model performs on real-world data. These metrics provide valuable insights into the model's strengths and weaknesses.

Detecting Data Leakage: Data leakage is a common issue where information from the testing data unintentionally influences the model's training. By keeping the testing set

separate from the training set, you can prevent data leakage and ensure that the model's performance evaluation is unbiased.

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

   I have used Linear Regression.

   Linear regressions can be used in business to evaluate trends and make estimates or forecasts. For example, if a company's sales have increased steadily every month for the past few years, by conducting a linear analysis on the sales data with monthly sales, the company could forecast sales in future months.

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

   Hyperparameter tuning is the process of finding the best set of hyperparameters (configurations not learned from data) for a machine learning algorithm to optimize its performance. It's important because the right hyperparameters can significantly impact a model's accuracy and generalization.

   One technique for hyperparameter tuning is "Grid Search," where you specify a range of values for each hyperparameter, and the algorithm systematically evaluates all combinations to find the best one. Benefits include simplicity and thorough exploration of hyperparameter space.