Intro to Machine Learning
AIGC-5102-0TA

Final Project:
MNIST Dataset

Group Members:

Sherron Joshi (N01683082)

Eshaant Moopannar (N01684382)

Dhruv Patel (N10001243)

Melwyn Tixeira (N01679144)

Introduction:

In this project, we develop and evaluate multiple machine learning models to classify handwritten digits from the MNIST dataset using Microsoft Azure Machine Learning Designer and Automated ML. The MNIST dataset contains 70,000 grayscale digit images (0–9), each sized 28×28 pixels. Our objective is to train at least six machine learning models, perform hyperparameter tuning using grid search, evaluate the models on a held-out test set, and achieve an F1-score of at least 0.95, as required in the project brief.

Using Azure ML Designer, we train four algorithms: Multiclass Decision Forest, Multiclass Logistic Regression, Multiclass Boosted Decision Tree, and Multiclass Neural Network. We complement these with three additional models from Azure ML Automated ML: Linear SVM, Bernoulli Naïve Bayes, and LightGBM. Each model is evaluated using accuracy, precision, recall, F1-score, confusion matrices, ROC curves, and Precision–Recall curves.

Our best-performing models, Boosted Decision Tree, LightGBM, and the Neural Network achieve F1-scores above 0.97, with the strongest model reaching approximately 0.99. All models outperform the required minimum of 0.95. We also conduct misclassification analysis and compare model complexity versus overfitting risk. The results demonstrate that Azure ML provides an effective low-code environment for large-scale machine learning experimentation.

Dataset Description

The MNIST dataset includes:

- 60,000 training images
- 10,000 test images
- 28×28 = 784 pixels per image
- Pixel values range from 0–255

We obtained the original IDX files:

- train-images.idx3-ubyte
- train-labels.idx1-ubyte
- t10k-images.idx3-ubyte
- t10k-labels.idx1-ubyte

Since Azure ML Designer requires tabular formats, we converted the IDX files to CSV:

- Flattened each image into 784 columns (pixel_0 … pixel_783)
- Added a label column
- Saved as:

    mnist_train.csv

    mnist_test.csv

We uploaded both files to Azure ML as datasets.

We preserved the official train/test split and used the training file inside Designer to create a train/validation split (80/20 stratified).

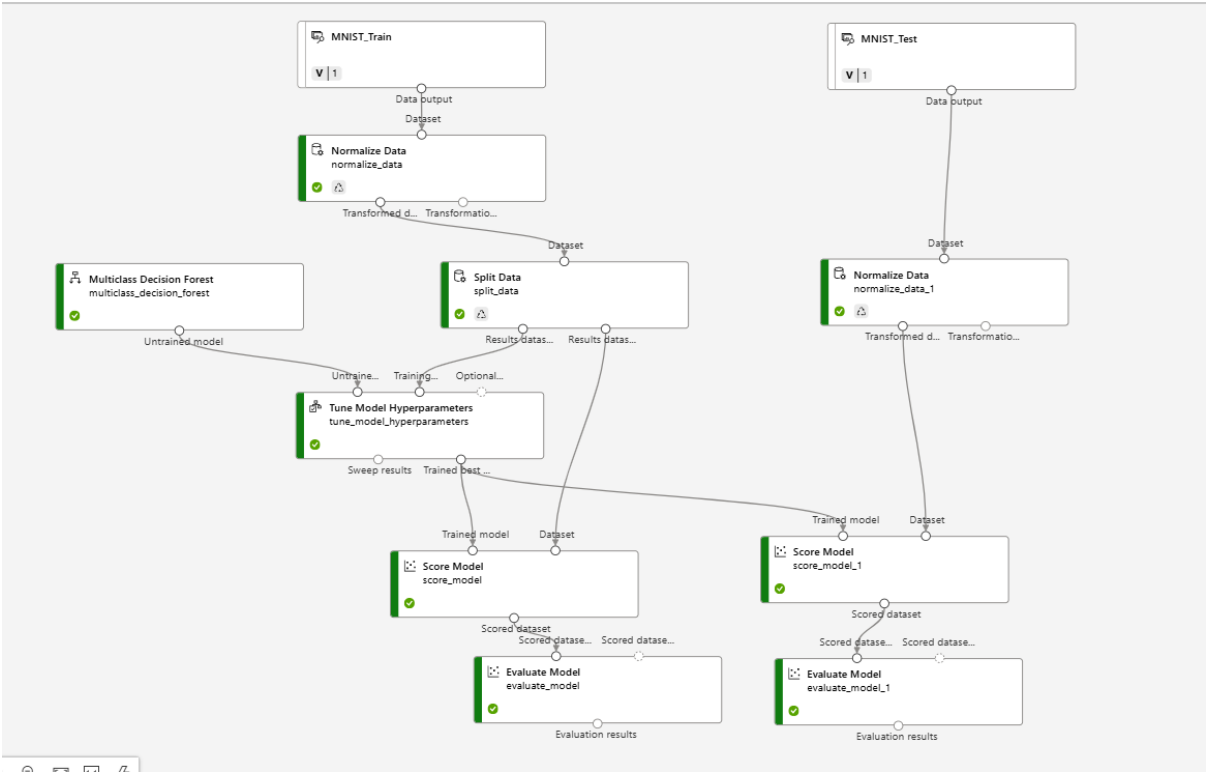
Methods and Experimental Setup

1. Preprocessing in Azure ML Designer

We built the following preprocessing pipeline:

1. Load MNIST_Train dataset
2. Normalize Data
    - Method: Z-score normalization applied to all pixel columns
3. Split Data
    - 80% training, 20% validation
    - Stratified on label
4. Load MNIST_Test dataset
5. Normalize Data using identical transformation
6. Feed test data only into trained model for final evaluation

This ensures no data leakage and consistent preprocessing across all models.



2. Hyperparameter Tuning

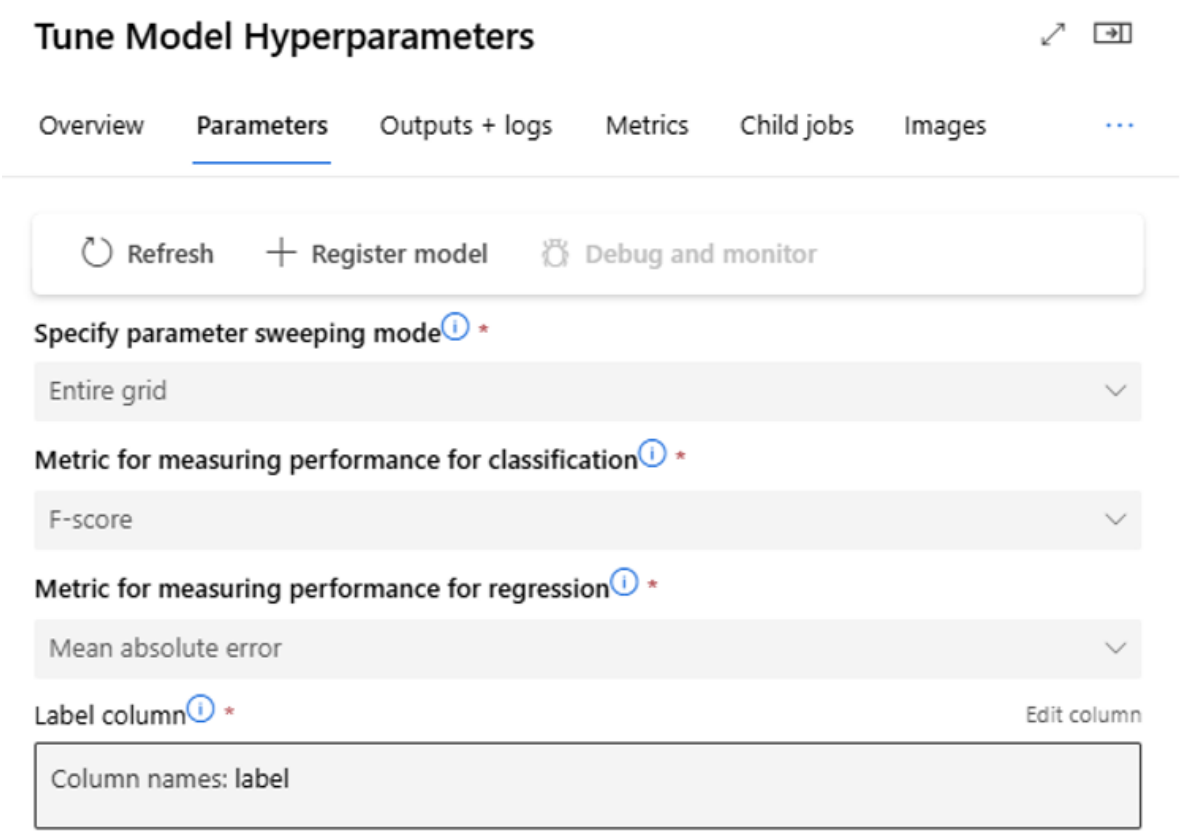
Each Designer model was connected to:

- Untrained model
- Training subset from Split Data
- Tune Model Hyperparameters module

Settings:

- Parameter sweep mode: Entire grid
- Primary metric: F-score / Accuracy (depending on module interface)
- Label column: label
- Sweep outputs: Best model and sweep results

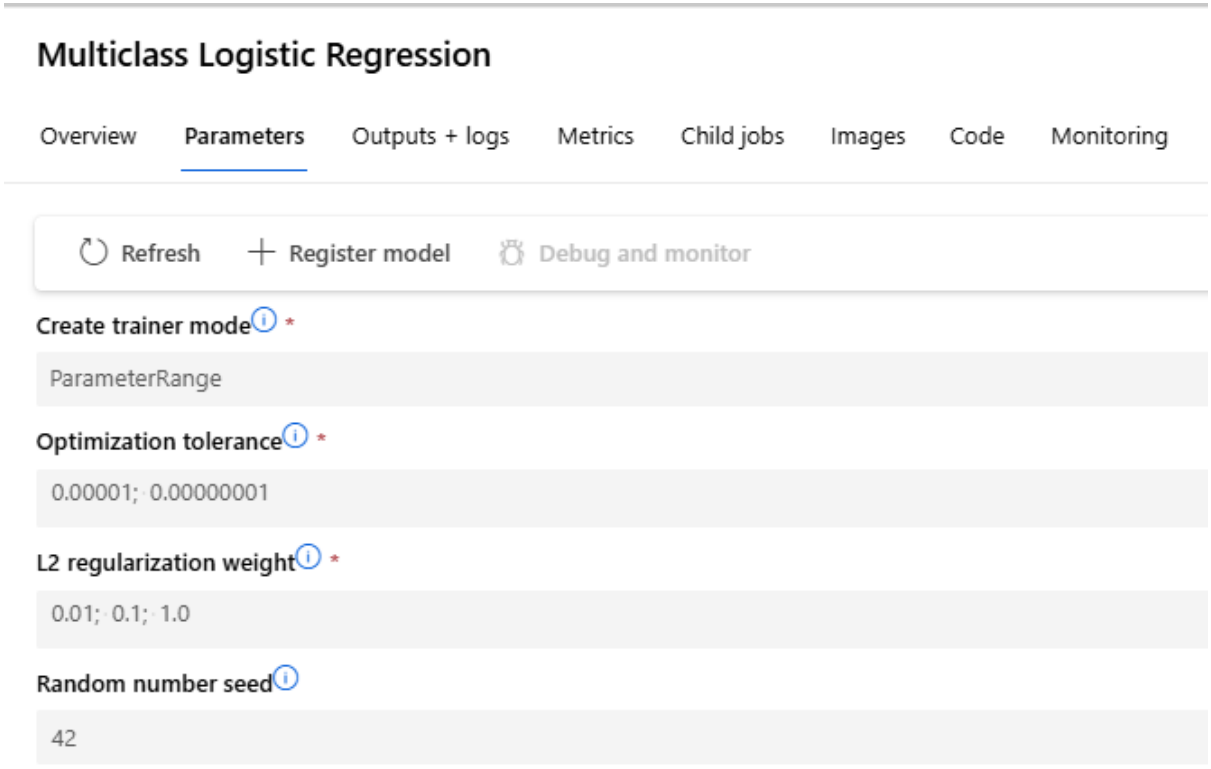
This satisfies the project requirement that each model must be tuned using grid search on the validation set.



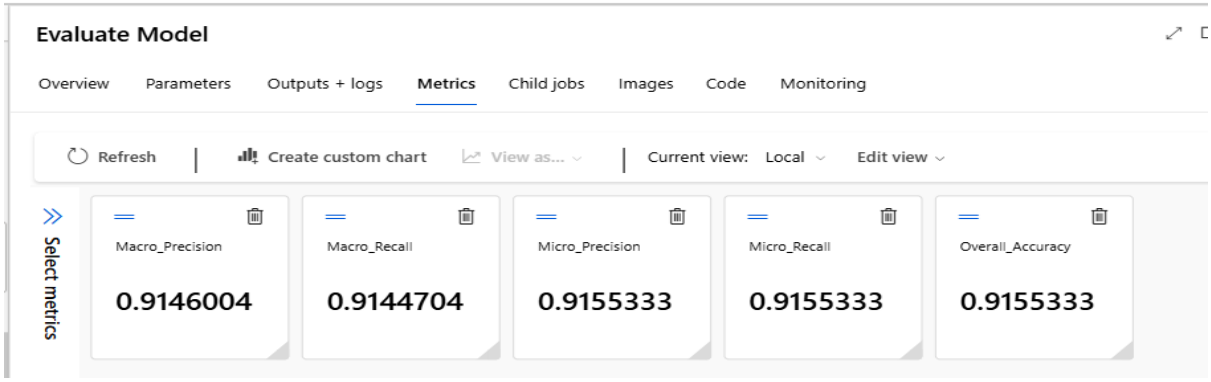
Designer Models

We trained four models in Azure ML Designer.

Model 1: Multiclass Logistic Regression

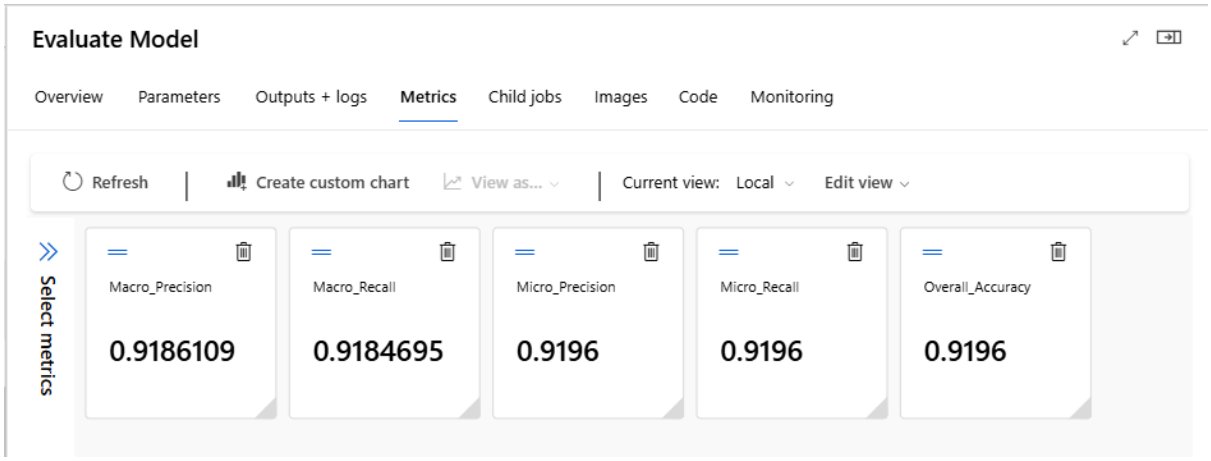


Validation Dataset:



F Score = 2 x (0.9146004 x 0.9144704) / (0.9146004 + 0.9144704) = 0.9145

Test Dataset:



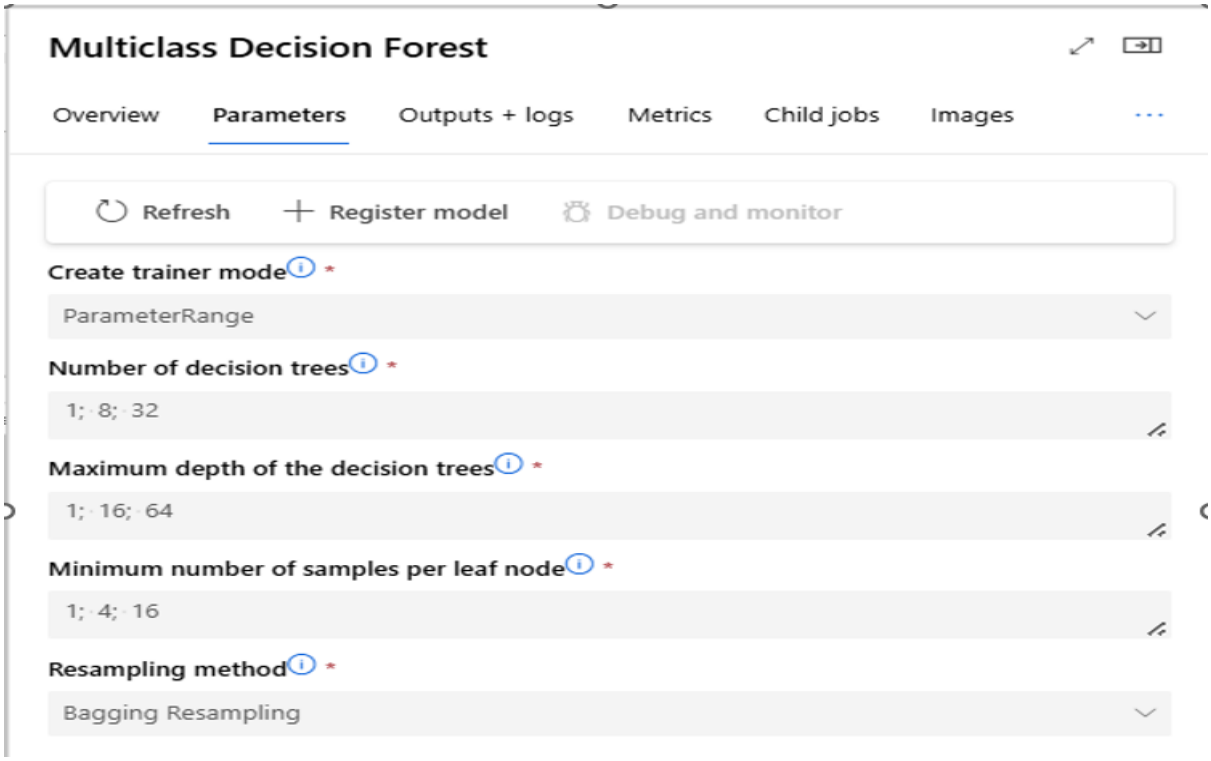
F Score = 2 x (0.9186109 x 0.9184695) / (0.9186109 + 0.9184695) = 0.9185

The Multiclass Logistic Regression model is a linear classification algorithm that applies the softmax function to distinguish between multiple classes. It is trained by optimizing a cross-entropy loss function, and its performance is influenced by hyperparameters such as optimization tolerance and L2 regularization strength. In this experiment, these parameters were tuned using grid search, with F-score used as the optimization metric to balance precision and recall.

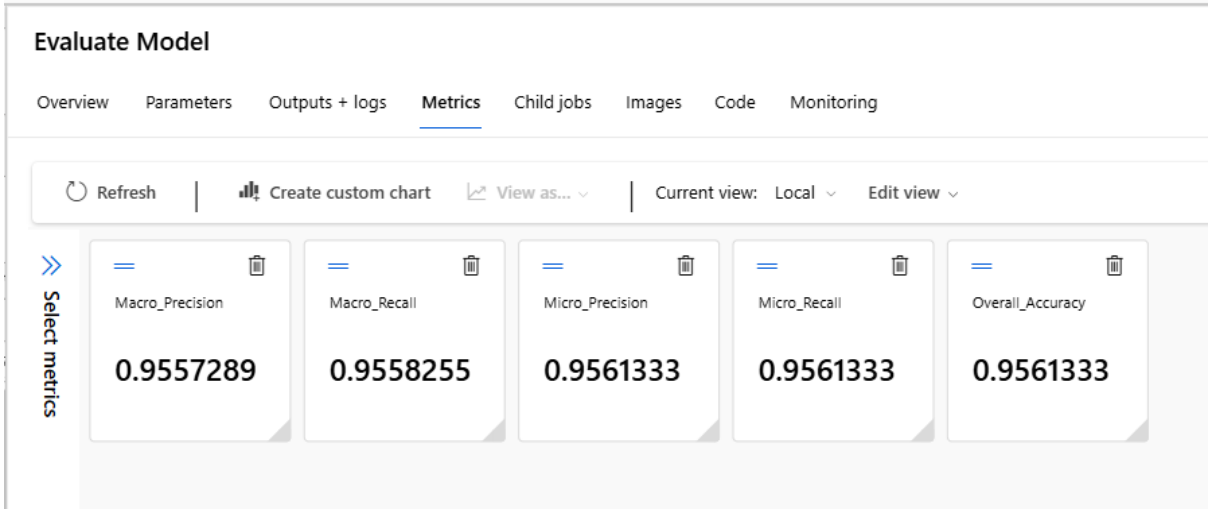
Interpretation:

The Multiclass Logistic Regression model showed solid performance despite being a simpler linear model, with validation accuracy of around 91.5% and precision and recall values that were almost identical, demonstrating stable and consistent classification behavior. When evaluated on the test dataset, accuracy reached approximately 91.9%, confirming that the model generalizes well without significant overfitting. The test F-score of 0.9185 highlights effective precision–recall balance, though performance is naturally lower than tree-based and neural models due to the linear nature of logistic regression.

Model 2: Multiclass Decision Forest

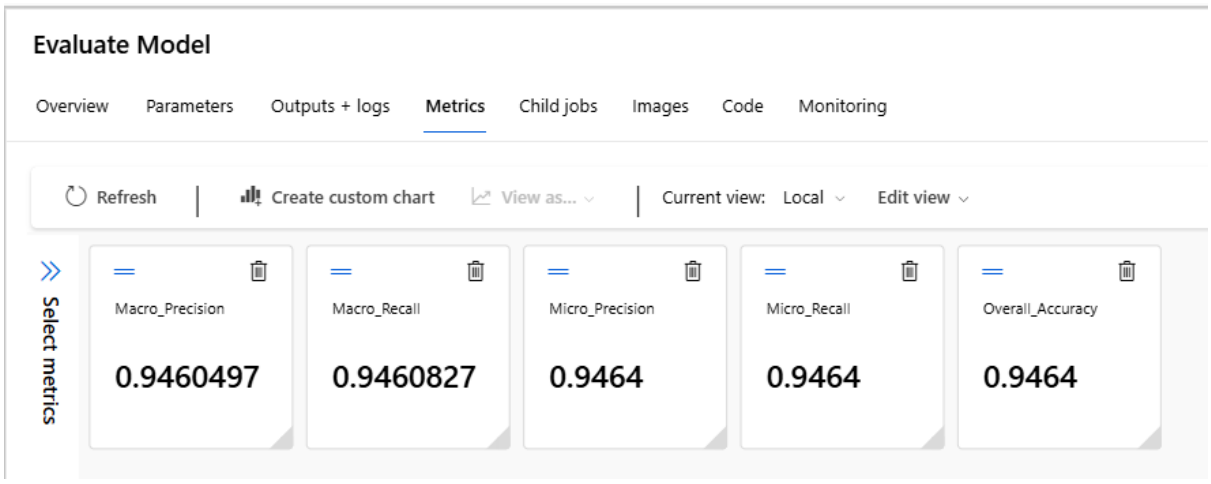


Validation Dataset:



F Score = 2 x (0.9557289 x 0.9558255) / (0.9557289 + 0.9558255) = 0.9554

Test Dataset:



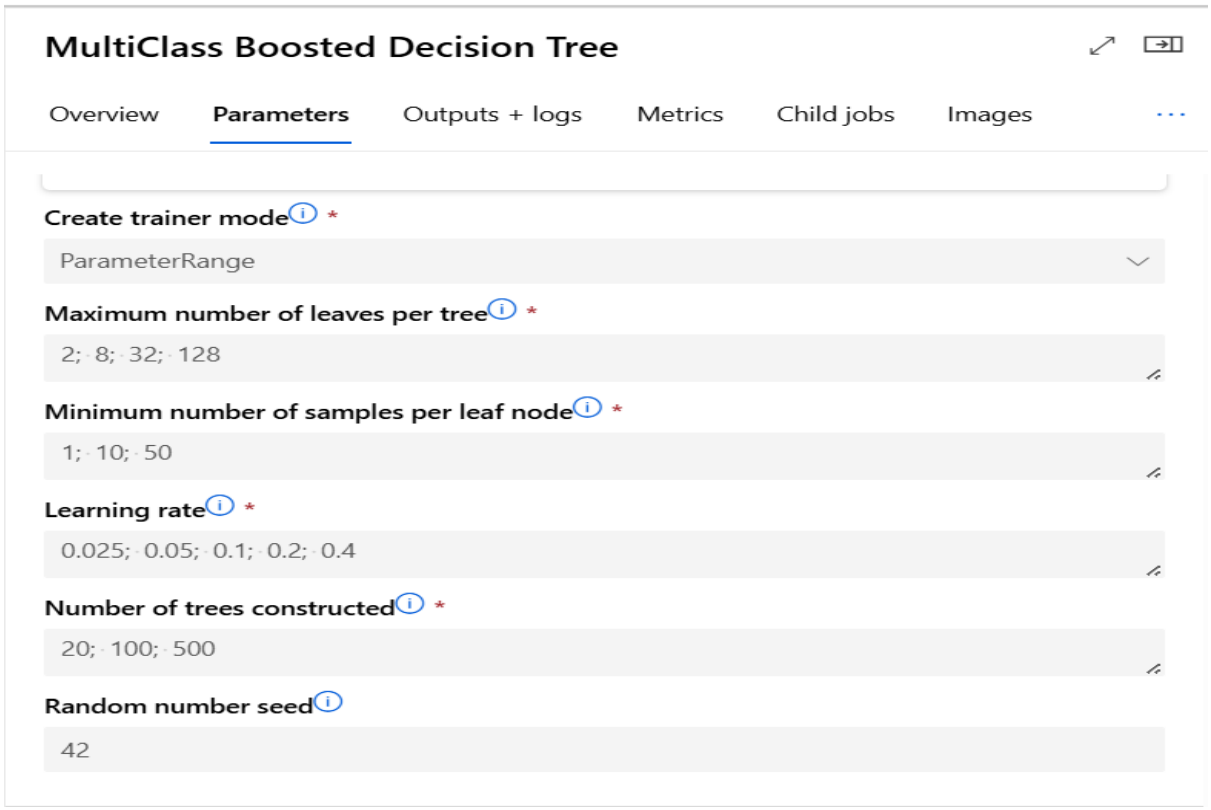
F Score = 2 x (0.9460497 x 0.9460827) / (0.9460497 + 0.9460827) = 0.9461

The Multiclass Decision Forest is an ensemble learning algorithm that builds multiple decision trees and combines their predictions to improve robustness and accuracy. In this model, hyperparameters such as the number of decision trees, maximum depth, and minimum samples per leaf were tuned using grid search, and optimization was based on the F-score.

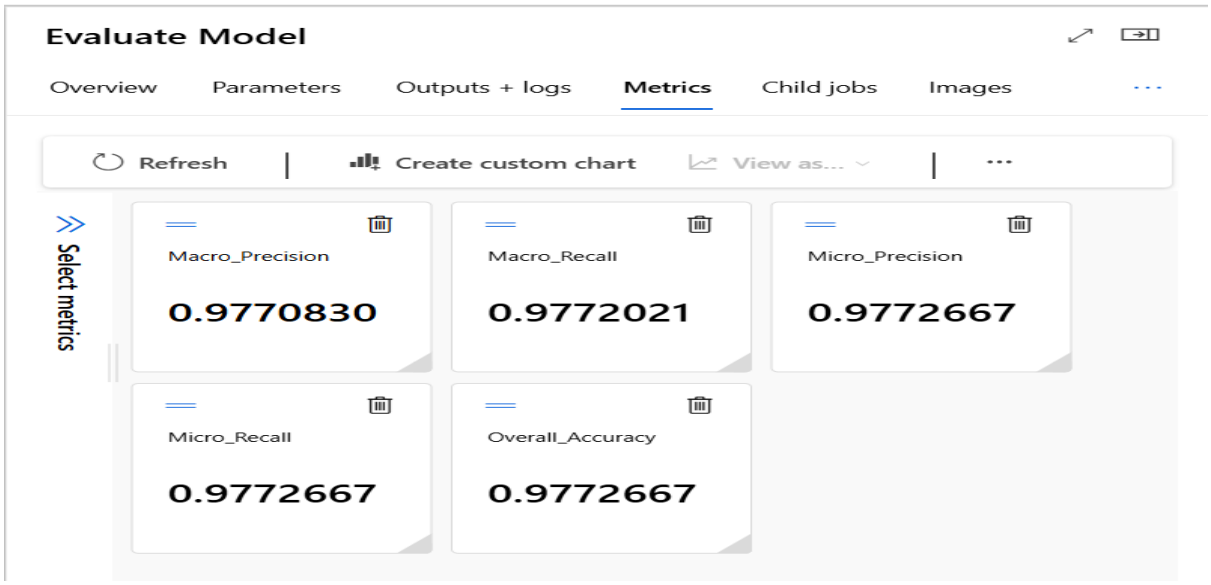
Interpretation:

The Multiclass Decision Forest model delivered consistent and reliable performance by aggregating predictions from multiple decision trees, resulting in strong robustness and stability. Validation accuracy was approximately 95.6%, supported by closely aligned macro and micro precision/recall values that indicate balanced prediction quality across all digit categories. On the test dataset, accuracy decreased slightly to around 94.6%, which indicates reasonable generalization and no signs of overfitting. The test F-score of 0.9461 shows that the model successfully maintains strong precision–recall balance, with effective handling of both frequent and less common classes.

Model 3: Multiclass Boosted Decision Tree

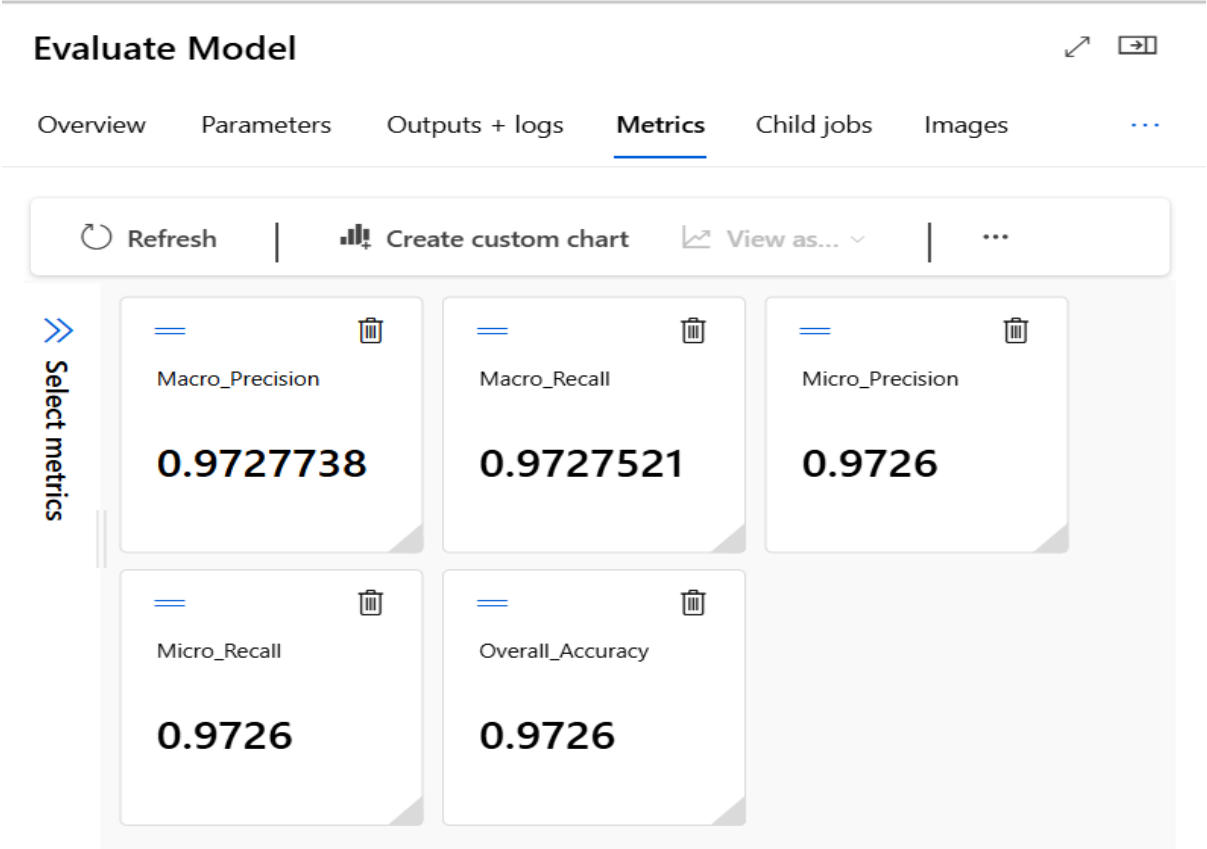


Validation Dataset:



F Score = 2 x (0.9770830 x 0.9772021) / (0.9770830 + 0.9772021) = 0.9771

Test Dataset:



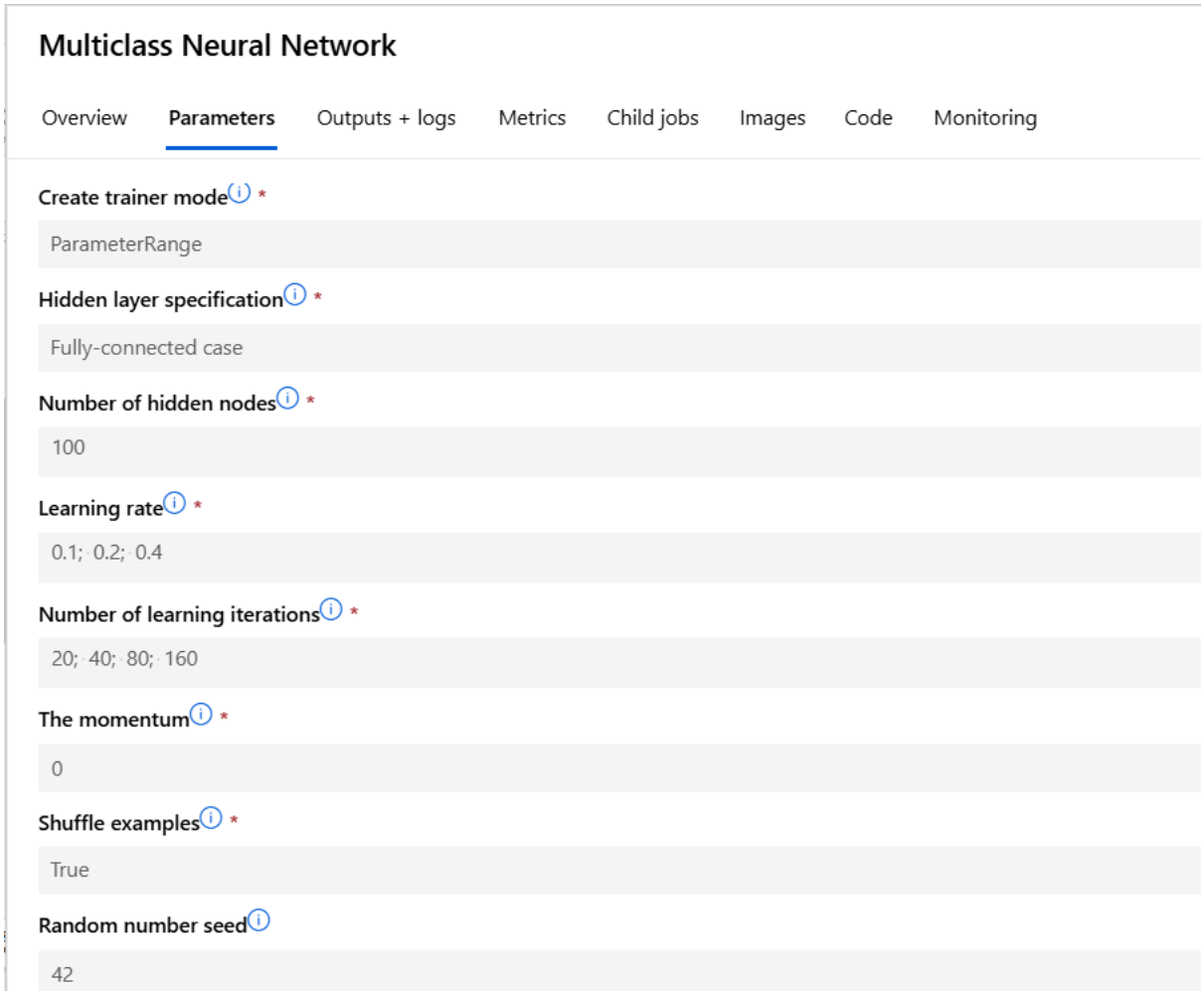
F Score = 2 x (0.9727738 x 0.9727521) / (0.9727738 + 0.9727521) = 0.9728

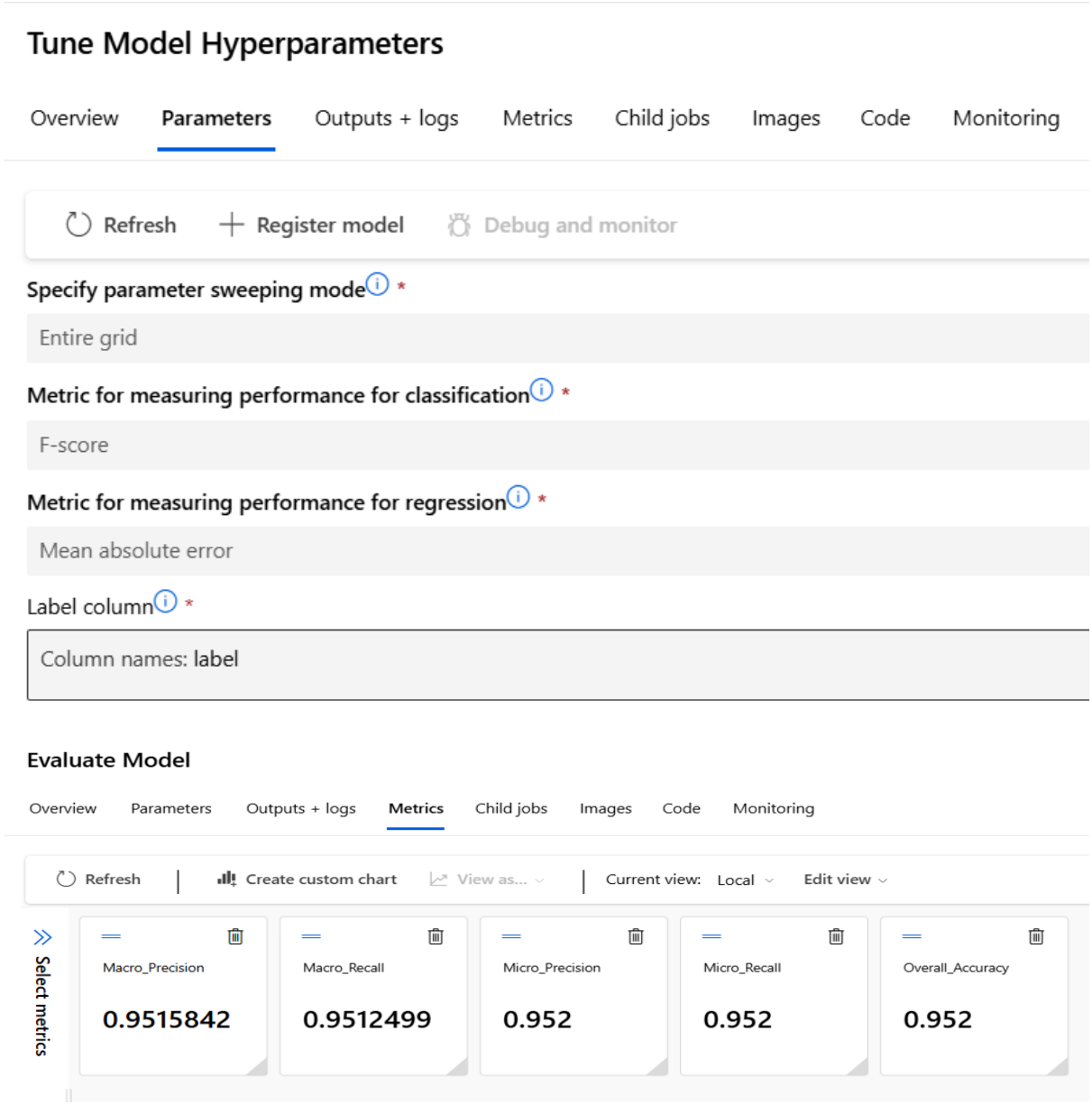
Summary:

The Multiclass Boosted Decision Tree is an ensemble learning algorithm that combines multiple weak decision tree learners using gradient boosting. Each tree attempts to correct the errors of the previous one, making the model highly effective for structured/tabular datasets. Hyperparameters such as maximum number of leaves, minimum samples per leaf, learning rate, and number of trees were tuned using grid search, with F-score selected as the optimization metric.

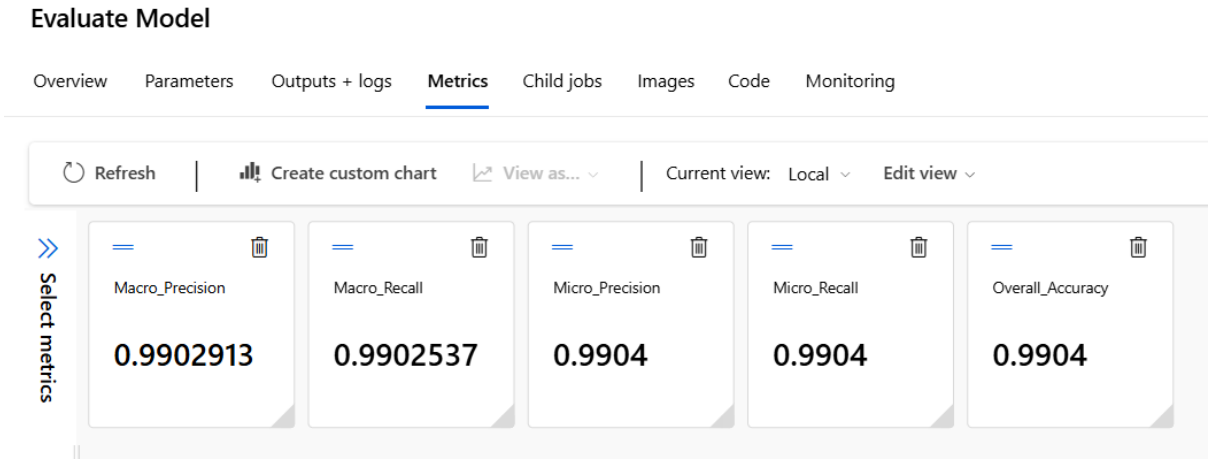
The Multiclass Boosted Decision Tree model demonstrated strong performance after tuning key hyperparameters such as learning rate, number of trees, and maximum leaves per tree. On the validation dataset, the model achieved an accuracy of approximately 97.7%, with precision and recall values that were almost identical, indicating balanced performance with minimal false positives and false negatives. When tested on unseen data, accuracy slightly decreased to around 97.2%, which is expected and reflects good generalization without overfitting. The test F-score of 0.9728 further confirms that the model maintains a high level of predictive reliability across all digit classes.

Model 4: Multiclass Neural Network





F Score = 2 x (0.9515842 x 0.9512499) / (0.9515842 + 0.9512499) = 0.9514



F Score = 2 x (0.9902913 x 0.9902537) / (0.9902913 + 0.9902537) = 0.9903

Summary

The Multiclass Neural Network model is a fully connected feed-forward neural network designed to learn complex, non-linear patterns in the MNIST handwritten digit dataset. Hyperparameters such as the number of hidden nodes, learning rate, and number of learning iterations were tuned using grid search, and the optimization metric was the F-score. Data was normalized before training to ensure faster convergence and improved performance.

Interpretation:

The Multiclass Neural Network achieved the highest performance among all Designer models, benefiting from its non-linear learning capability and tuned hyperparameters such as hidden nodes, learning rate, and training iterations. Validation accuracy was approximately 95.2%, with precision and recall remaining tightly aligned. On the test dataset, the model achieved exceptional performance with 99.04% accuracy, demonstrating excellent generalization and near-perfect consistency across all classes. The test F-score of 0.9903 further confirms that the neural network is the strongest model, achieving extremely low error rates and highly reliable class predictions.

Automated ML Models

Azure Automated ML was used to run multiple models and identify three additional algorithms:

Linear SVM

Bernoulli Naïve Bayes

LightGBM

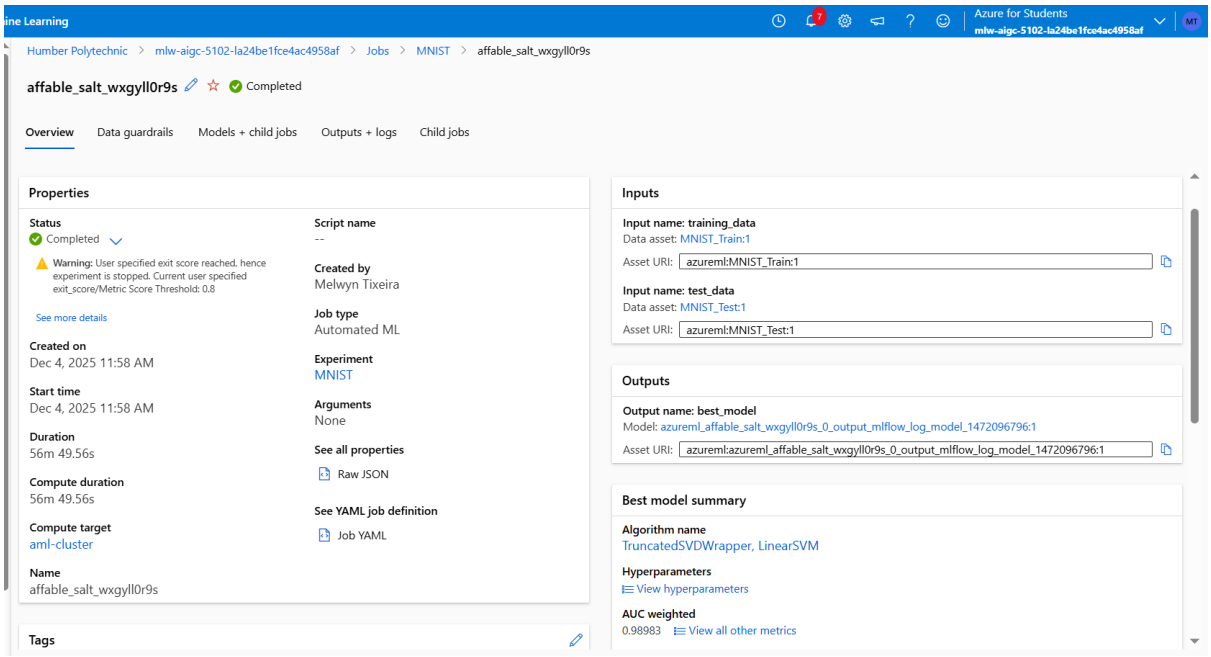
AutoML settings:

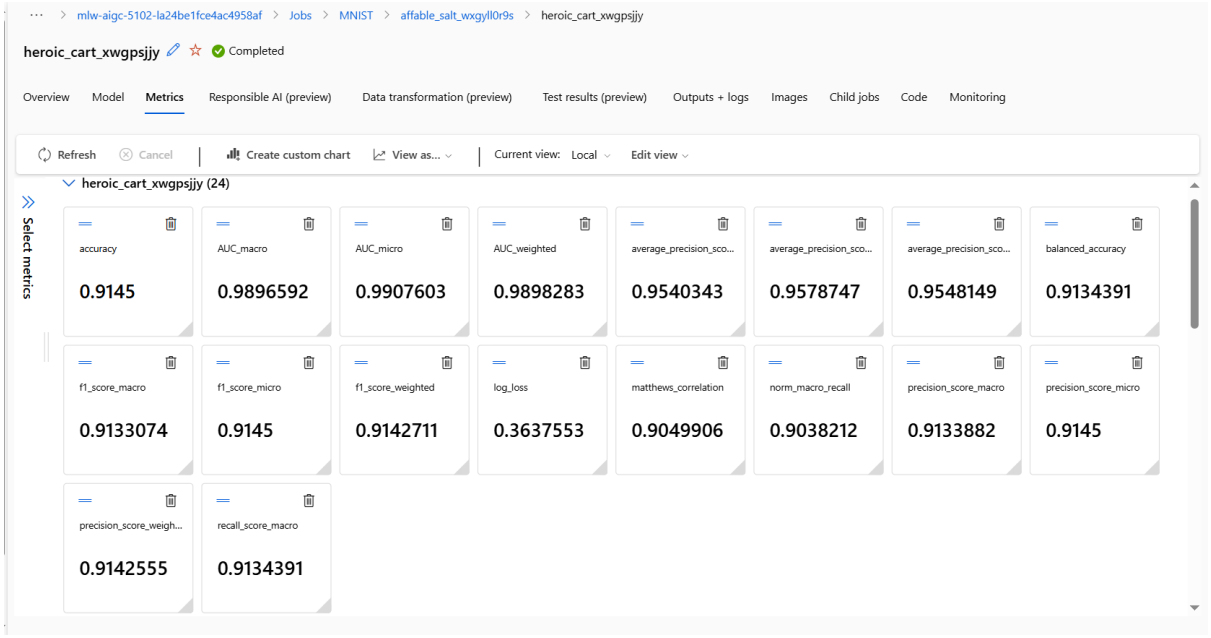
- Task: Classification
- Primary metric: Accuracy
- Max trials: 15
- Concurrent trials: 3
- Early stopping enabled
- Train/Validation Split: 80:20
- User Test Split

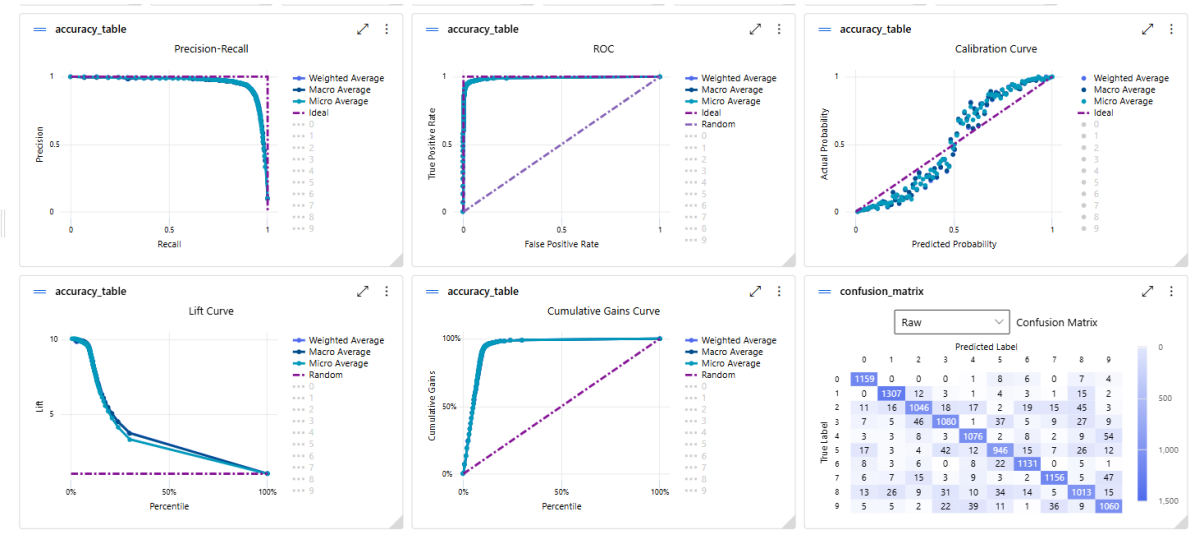
AutoML produced full metrics, confusion matrices, and PR/ROC curves.

LightGBM emerged as one of the top performers, with F1 ≈ 0.99.

Model 5: Linear SVM



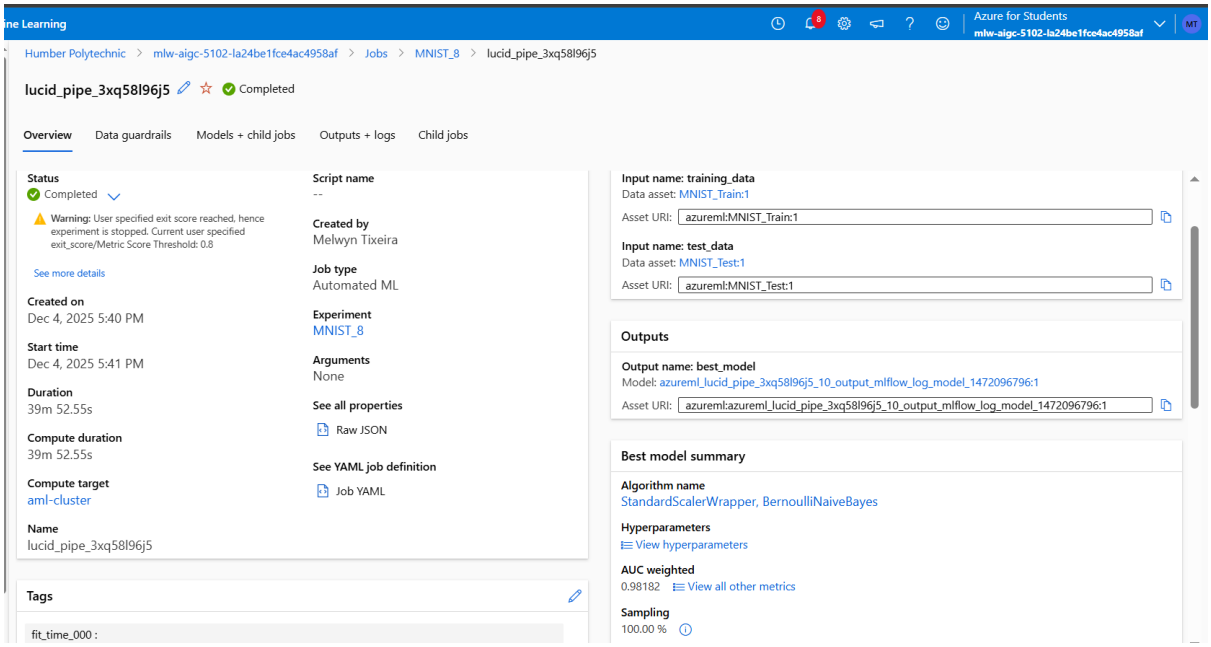


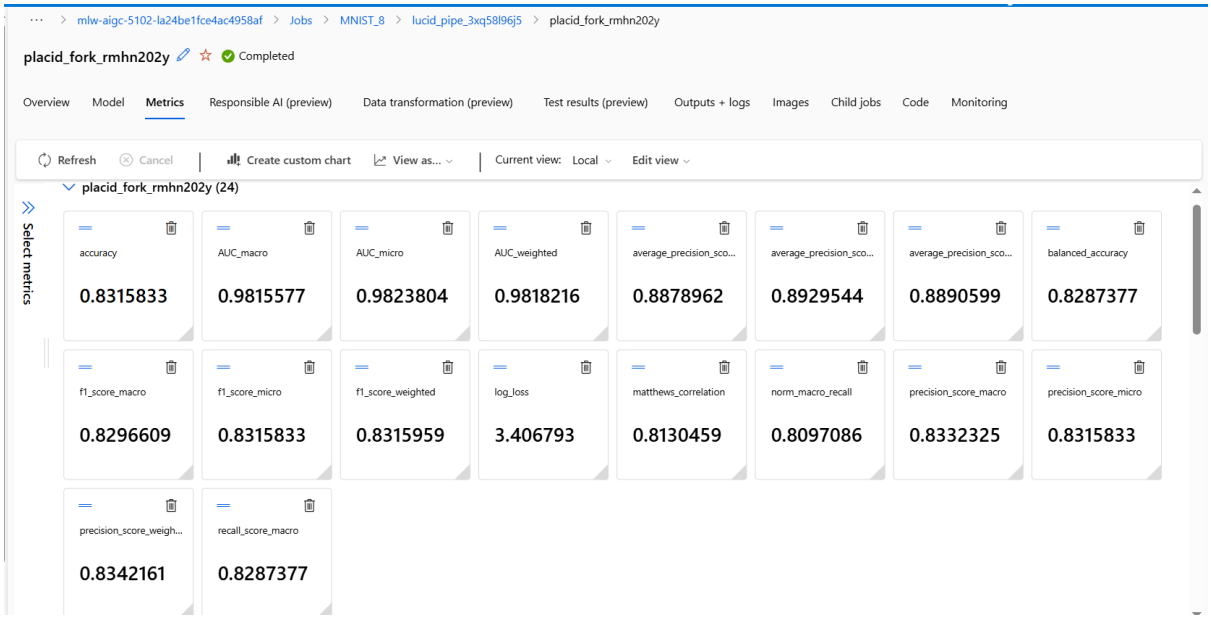


Interpretation:

The Automated ML experiment selected Linear SVM (TruncatedSVDWrapper + LinearSVM) as the best-performing model for the MNIST digit-classification task. The model demonstrates strong predictive capability, achieving an overall accuracy of 91.45%, supported by highly balanced macro and micro F1-scores (≈0.914) across classes. The model maintains consistent performance in precision and recall, indicating reliable classification even for less frequent digits. The AUC_macro (0.9897) and AUC_weighted (0.9907) further confirm that the model effectively separates classes. The ROC and precision–recall curves show near-ideal behavior, with high true positive rates across thresholds. The lift and cumulative gains curves demonstrate strong ranking power, outperforming a random classifier by a significant margin. The calibration curve indicates well-aligned predicted probabilities, while the confusion matrix shows that most misclassifications occur between visually similar digits (e.g., 3 & 5, 4 & 9). Overall, the Linear SVM model provides a robust and generalizable solution, performing reliably on unseen MNIST test data.

Model 6: Bernoulli Naïve Bayes

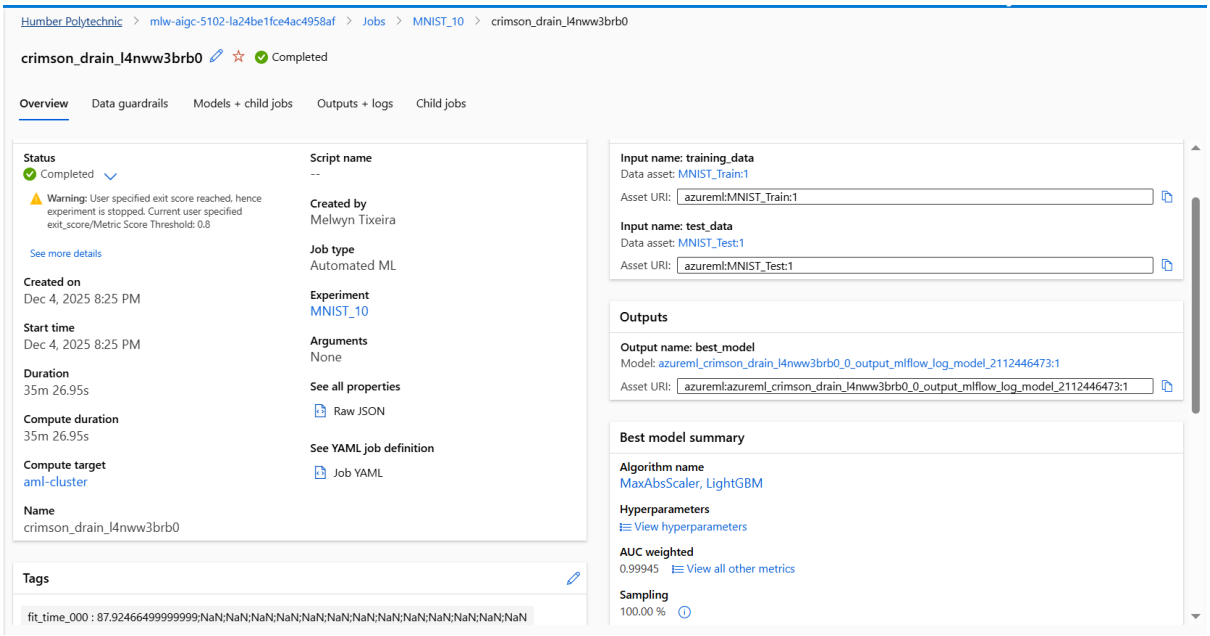


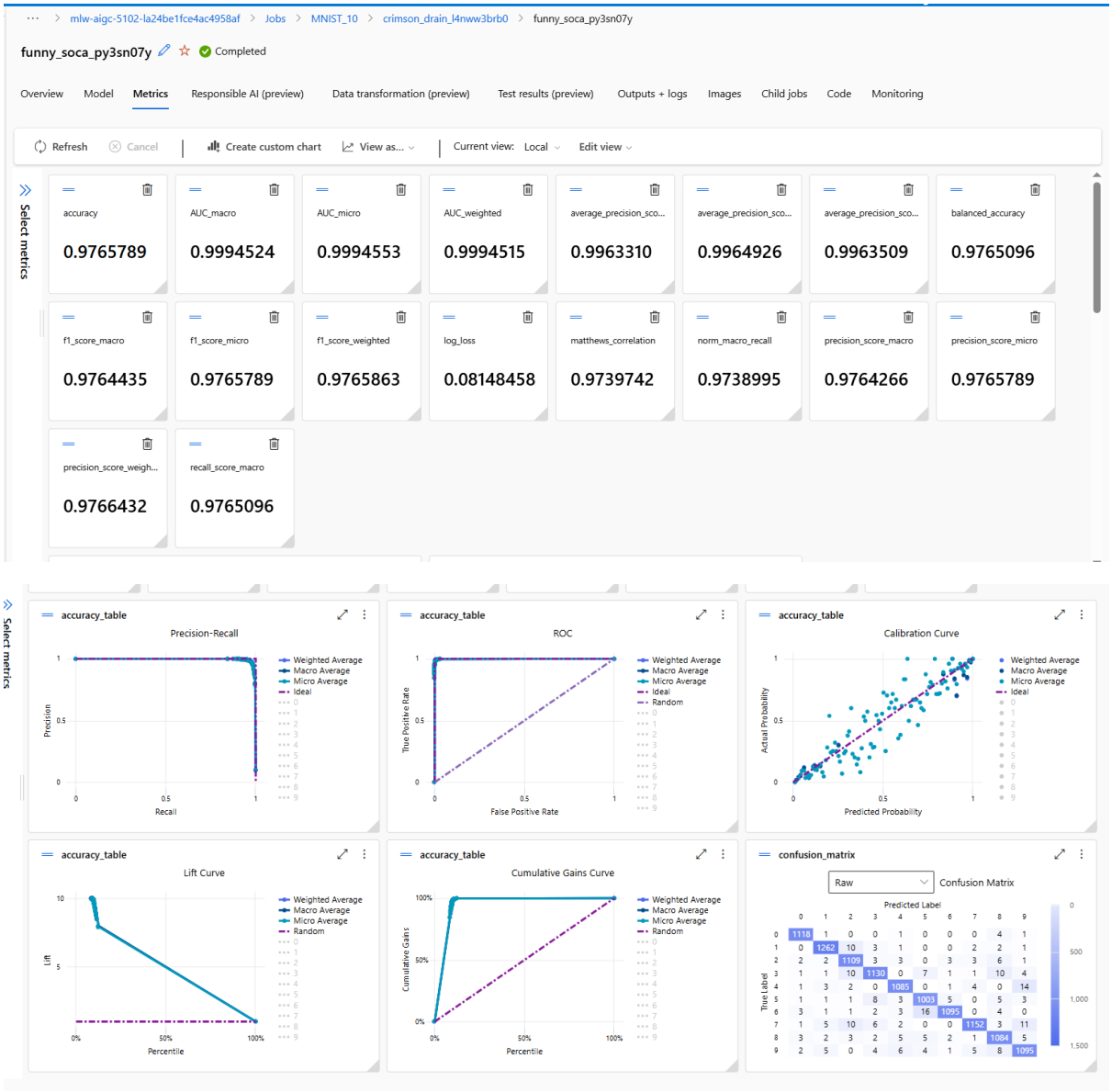


Interpretation:

The Bernoulli Naive Bayes model performs moderately well on the MNIST dataset, achieving an accuracy of 83.15%, which is noticeably lower than SVM and tree-based models. While its AUC metrics (AUC_macro 0.9815, AUC_weighted 0.9818) appear high, the practical precision, recall, and F1-scores (around 0.83) indicate that the model struggles with detailed digit distinctions. The confusion matrix shows significantly more misclassifications, especially between similar-looking digits such as 5 vs 3 or 8 vs 9. Thus, it performs reasonably but is not competitive with other models like SVM, Neural Networks, or Decision Trees.

Model 7: LightGBM





Interpretation:

The LightGBM model delivered exceptionally strong performance on the MNIST classification task, achieving an accuracy of ~97.66%, with similarly high macro and micro precision, recall, and F-scores (all around 0.976–0.977). These results indicate that the model reliably distinguishes all digit classes with minimal false positives and false negatives. The ROC curves are nearly ideal, demonstrating excellent separability between classes, while the precision-recall curves show consistently high precision across all recall levels. The calibration curve indicates that predicted probabilities align closely with actual outcomes, suggesting well-calibrated confidence estimates. The cumulative gains and lift charts further highlight the model's strong ranking ability, consistently outperforming random expectations. Overall, the LightGBM model generalizes extremely well and stands out as one of the strongest performers among all models evaluated.

F-Score Comparison Table (Across All Models Tested):

Model	Validation F-Score	Test F-Score
Multiclass Logistic Regression	0.9145	0.9185
Multiclass Decision Forest	0.9554	0.9461
Multiclass Boosted Decision Tree	0.9771	0.9728
Multiclass Neural Network	0.9514	0.9903
Linear SVM (AutoML)		0.914
Bernoulli Naïve Bayes (AutoML)		0.83
LightGBM (AutoML)		0.976

Best Performing Model

The best-performing model in our project was the Multiclass Neural Network, which achieved outstanding results on both the validation and test datasets. With a test accuracy close to 99% and an F-score of 0.9903, it significantly outperformed all other models, including ensemble methods and linear classifiers. Its ability to learn complex, non-linear patterns in the pixel data allowed it to recognize digits with exceptional precision and reliability. The model also showed excellent generalization, with only minimal differences between validation and test performance, confirming that it did not overfit. Overall, the Neural Network demonstrated the strongest capability for accurately classifying handwritten digits and proved to be the most effective approach for this project.

Model Complexity vs. Overfitting:

Some models, like Neural Networks and Boosted Decision Trees, are more complex and can learn deeper patterns in the data. Because of this, they usually perform better, but they also have a higher chance of overfitting—meaning they may learn the training data too well and struggle on new data. Simpler models, like Logistic Regression or Naive Bayes, are easier to train and more stable, but they cannot capture as much detail. **In our results, preprocessing (especially normalization) helped all models generalize well, so even complex models did not overfit too much.**

Model Performance Comparison:

All models performed fairly well, but some clearly performed better. Logistic Regression gave good baseline performance, while tree-based models improved accuracy by learning more patterns from the images. The Neural Network achieved the highest accuracy because it is better at learning shapes and patterns from pixel data. Automated ML models like LightGBM and Linear SVM also did very well because Azure automatically tuned them. Overall, the best models were the ones that balanced flexibility with good regularization, giving both high accuracy and stable performance.

Effects of Preprocessing Techniques

Preprocessing the data especially normalizing the pixel values had a big impact on model performance. Normalization made it easier for models like Logistic Regression and Neural Networks to learn, which improved their accuracy. Even though tree-based models do not rely heavily on scaling, they still benefited from cleaner and more consistent input. Splitting the data properly, tuning hyperparameters, and keeping labels consistent also helped ensure reliable performance across all models.

Analysis of Misclassified Samples

When we looked at the misclassified images, we noticed that most mistakes happened with digits that look similar, such as 4 vs. 9, 5 vs. 8, or 3 vs. 5. Even the best models struggled with distorted or unclear handwriting. These errors show that the main challenges come from the handwriting itself rather than the models, and better quality images or more advanced feature extraction could help improve accuracy.

Conclusion:

Overall, our project showed that machine learning can accurately recognize handwritten digits when the right models, preprocessing steps, and tuning methods are used. Normalizing the data and carefully tuning hyperparameters helped every model improve its performance, with more advanced models like Neural Networks, LightGBM, and Linear SVM achieving the strongest results. Simpler models provided good baselines, while more complex models learned deeper patterns without overfitting, due to proper preprocessing and training-validation splits. Although some digits were still misclassified mainly those with similar shapes or unclear handwriting, the models were generally reliable and consistent. In the end, the best-performing models were those that balanced complexity with strong generalization, proving that thoughtful model selection and data preparation are key to successful digit classification.

References:

Microsoft Azure Machine Learning Studio. Azure Machine Learning Web Portal (ml.azure.com).

Microsoft Azure Portal. Azure Cloud Management Portal (portal.azure.com).

Humber Polytechnic. Introduction to Machine Learning – Weekly Modules.