**CST8921 – Cloud Industry Trends**

**Lab 1 – AI and ML**

# Introduction

In this lab, I explored Azure machine learning and Azure machine learning studio to learn how to create and deploy machine learning models without involving complicated procedures or code. Azure machine learning allows three ways to generate an ML model: Notebooks, Automated ML, and Designer. I used the designer to create the Automobile Price Prediction (Basic) sample Regression model template for the lab. The designer uses a training pipeline to clean, train and evaluate the data input to the model.

# Objective

The goal of this lab activity is to familiarize myself with Azure ML, create a sample automobile model using an ML pipeline, and evaluate the model’s accuracy by comparing the predicted automobile pricing output to the actual automobile pricing.

# Steps Covered With Screenshots

### Creating Azure Machine Learning in Canada Central

### Using Public for Networking (For ease of use)

### The ML Review

### The Created ML Workspace, launching ML Studio

### Exploring within ML Studio

### The ML Training Pipeline- Selecting Automobile Price Prediction

### Creating a new experiment and setting the name as **train-regression-designer-ml**

### Using Defaults on Inputs and Outputs.

### Here is the Compute instance error, we need to create a new Compute instance to fix.

### Creating A Compute instance using Standard DS11\_v2 CPU.

### Everything else is default settings this is the Review Page

### Waiting for the Compute instance to create and start running: Below is confirmation of the instance running

### Reviewing and Executing the pipeline

### The pipeline has been executed and it now starts running the workloads

### Here is the jobs tab to review workloads clicking within shows the completion status.

### Here is the train-regression-designer-ml workload pipline completion status below it shows the completion of each task.

### Here is the list of Completed Tasks

### Here is a review on a specific completed workload below shows the date completed and the duration,

### Further Exploring, We can select the output of the cleaning dataset and below is the completed cleaned dataset output.

### Let’s explore Model Scoring to verify its integrity. Here we can view the predicted result prices as Scored Labels, and the actual price of the car as Price. You can see here, that the ML model tends to score the prices with marginal error with slightly higher or lower results as the actual price.

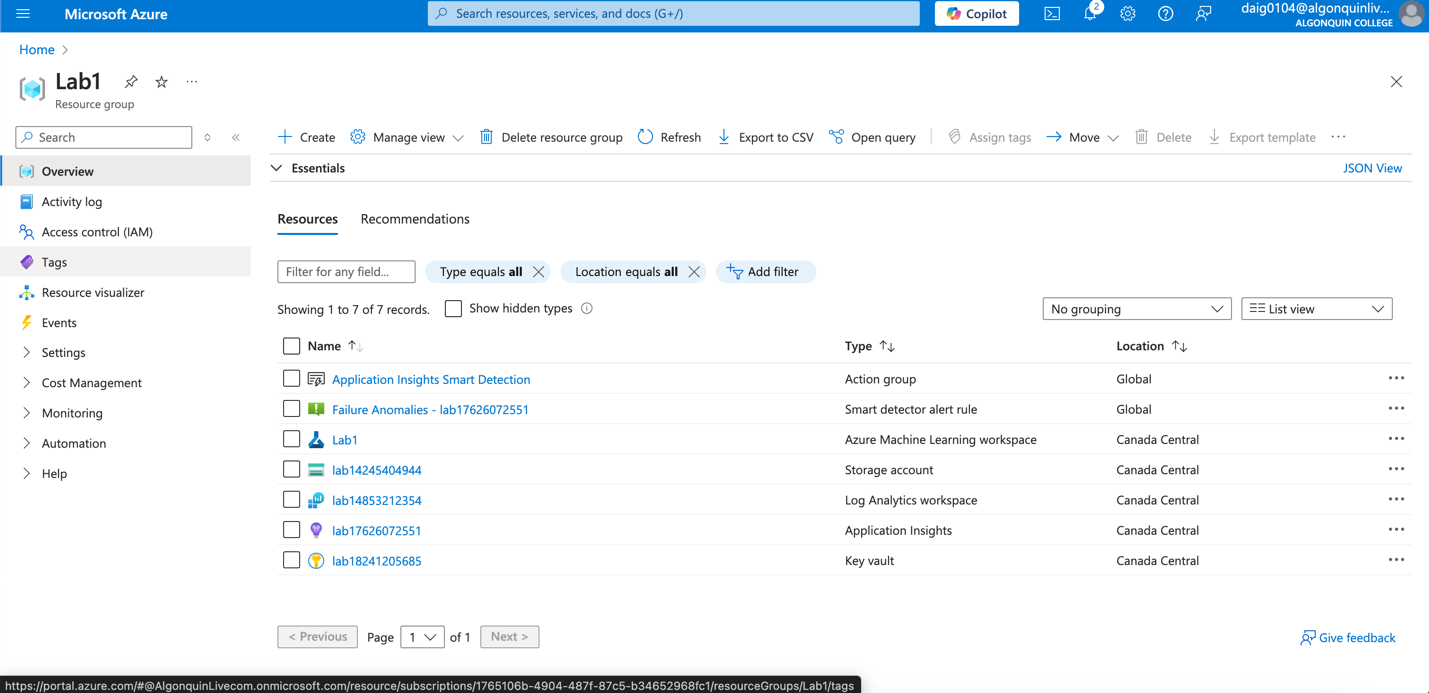
### Closer Look into the Scored Labels.

### Next is to check the Evaluation Model below, is the logs screen

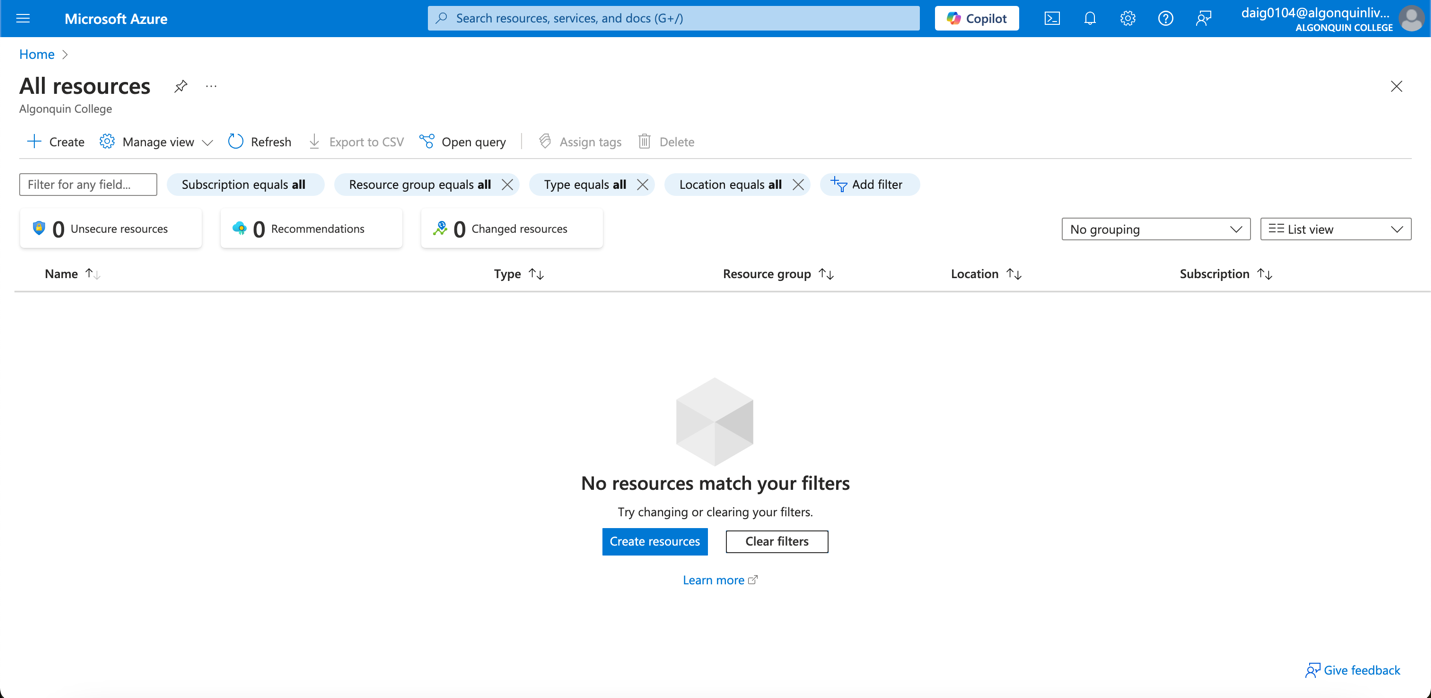
### And here is the Evaluation Model results:

### Here I wanted to go back to prove the high inaccuracies of the model as I wondered why the Mean\_Absolute\_Error is unusually high, and I have found this: Below you can see a price of a car being $5151 when the model predicted $-3335 (This is the same table as in step 21, just zoomed in for better viewing)

### Deleting all resources Before:



### After:



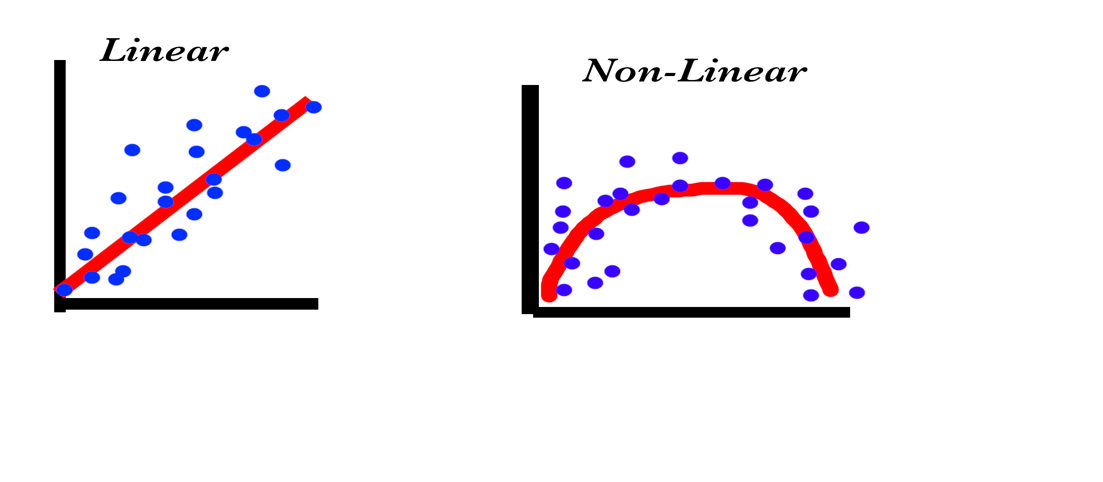
# Results:

The result is that the model has output a Scored\_dataset, a table of actual car prices called “Prices” and the car price predictions called “Scored Labels. The model has also outputted Evaluation results that use a numerical value to evaluate model accuracy. The resulting accuracy score as “Mean\_Absolute\_Error” is a high 1773.61.

# Conclusion:

In conclusion, the sample Automobile Pricing Regression model is not as accurate as expected. What I found is that the Mean\_Absolute\_Error value is quite high, 1773.61, which means that the average discrepancy in pricing has an error value of $1773.61 difference, where the value in theory should be much lower. A model is considered a good “fit” if the difference between the prediction and the real price remains minimal.   
  
Another discrepancy to the model is the prediction of $-3335 as a price value (step 24 Scored Labels), which proves prediction inaccuracy. This can also prove that the model, in theory, does not filter outputs, and a rule should be created to avoid negative prices as an output since it is not realistic in real-world scenarios. (what are the chances of a dealership owing you money?)

Given my knowledge of neural networks, there are many reasons why I believe the model may not be accurate. One possibility is the limitations of the hardware used to process the data. Neural Networks highly depend on how many neural “nodes” (think of them like brain neurons) process the data. The more nodes, the more accurate the data you may predict. In the case of this linear regression model, I chose a Standard\_DS11\_v2 CPU to process the data, which may limit the number of nodes and the accuracy of predictions compared to a GPU such as an ND\_MI300X\_v5-series.   
  
Another possibility is the data and the fit to the type of model created, linear regression models may not be the correct model to use as it depends upon the relation of the predicted data and the real value to be linear. Instead of linearity, the actual values plotted may be non-linear such as quadratic or logarithmic. In these cases, switching to a different algorithm that handles non-linearity yields better results. A solution to check for non-linearity of data is to plot the Scored\_Dataset Scored labels (the predictions) and Prices (the actual car price) within step 21 to a linear regression graph and check if the predicted line fits the real prices. Below is my rough drawing of linear data points vs non-linear data points.



The model should be changed when the data points (the blue dots) are Non-Linear, such as the graph on the right. In that case, linear predictions (the red line on the left) cannot fit correctly into the right graph, explaining the inaccuracy. (Microsoft, 2024) (Winkler, 2024)

# Works Cited

Microsoft. (2024, August 28). *Evaluate Model component*. Retrieved January 2025, from Microsoft Learn: https://learn.microsoft.com/en-us/azure/machine-learning/component-reference/evaluate-model?view=azureml-api-2#metrics-for-regression-models

Microsoft. (2024, November 19). *Sizes for virtual machines in Azure*. Retrieved January 2025, from Microsoft Learn: https://learn.microsoft.com/en-us/azure/virtual-machines/sizes/overview?tabs=breakdownseries%2Cgeneralsizelist%2Ccomputesizelist%2Cmemorysizelist%2Cstoragesizelist%2Cgpusizelist%2Cfpgasizelist%2Chpcsizelist

Sarvandani, M. (2023, June 9). *Top 11 algorithms of non-linear regression in machine learning + Proposed Python library+Python code*. Retrieved January 2025, from Medium: https://medium.com/@mohamadhasan.sarvandani/top-algorithms-of-non-linear-regression-in-machine-learning-proposed-python-library-and-python-c871752ddd81

Winkler, J. (2024, Febuary 14). *Machine Learning - Linear Regression Algorithm*. Retrieved January 2025, from LinkedIn: https://www.linkedin.com/pulse/machine-learning-linear-regression-7aasc