**DEPLOYING AZURE ML MODEL USING EMPLOYEE ATTRITION DATASET**

# INTRODUCTION AND OVERVIEW

The Synthetic Employee Attrition Dataset is a simulated dataset designed for the analysis and prediction of employee attrition. It contains detailed information about various aspects of an employee's profile, including demographics, job-related features, and personal circumstances.

The dataset comprises 74,498 samples, split into training and testing sets to facilitate model development and evaluation. Each record includes a unique Employee ID and features that influence employee attrition. The goal is to understand the factors contributing to attrition and develop predictive models to identify at-risk employees.

This dataset is ideal for HR analytics, machine learning model development, and demonstrating advanced data analysis techniques. It provides a comprehensive and realistic view of the factors affecting employee retention, making it a valuable resource for researchers and practitioners in the field of human resources and organizational development

**Purpose:** This document aims to guide stakeholders through the process of deploying a machine learning model trained on the Employee Attrition dataset using Azure Machine Learning.

**Audience:** Data scientists, developers, and operations teams involved in model deployment and maintenance.

# 2. SYSTEM ARCHITECTURE

**Diagrams:** Include a flowchart or architectural diagram showing how different components like Azure ML Workspace, Compute Instance and Model Deployment interact.

# 3. DEPLOYMENT ENVIRONMENT

**Hardware specifications:**

System Manufacturer – HP

Processor – 12th Gen Intel(R) Core(TM) i7-1255U, 1700 Mhz, 10 Core(s), 12 Logical Processor(s)

Hardware Abstraction Layer – Version= “10.0.22621.2506”

BIOS Version/Date – AMI F.19, 2023/07/03

RAM – 16.0 GB

Total Physical Memory – 15.7 GB

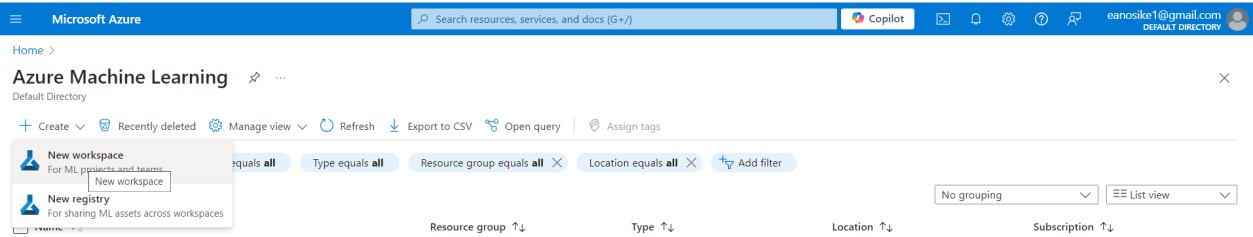
Total Virtual Memory – 32.6 GB

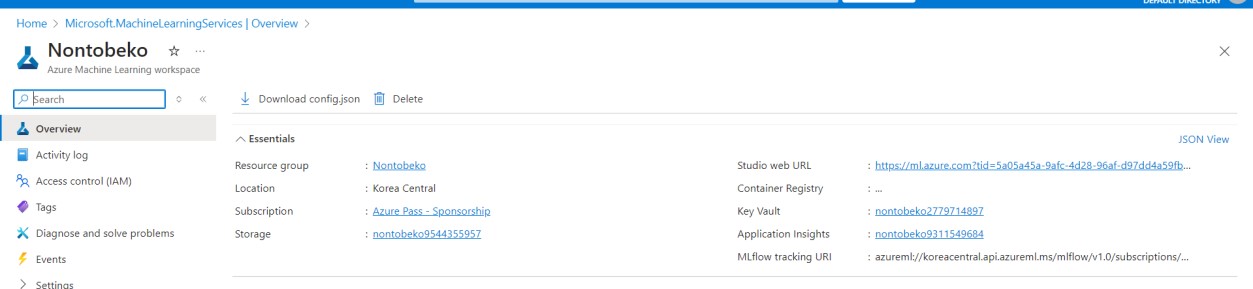
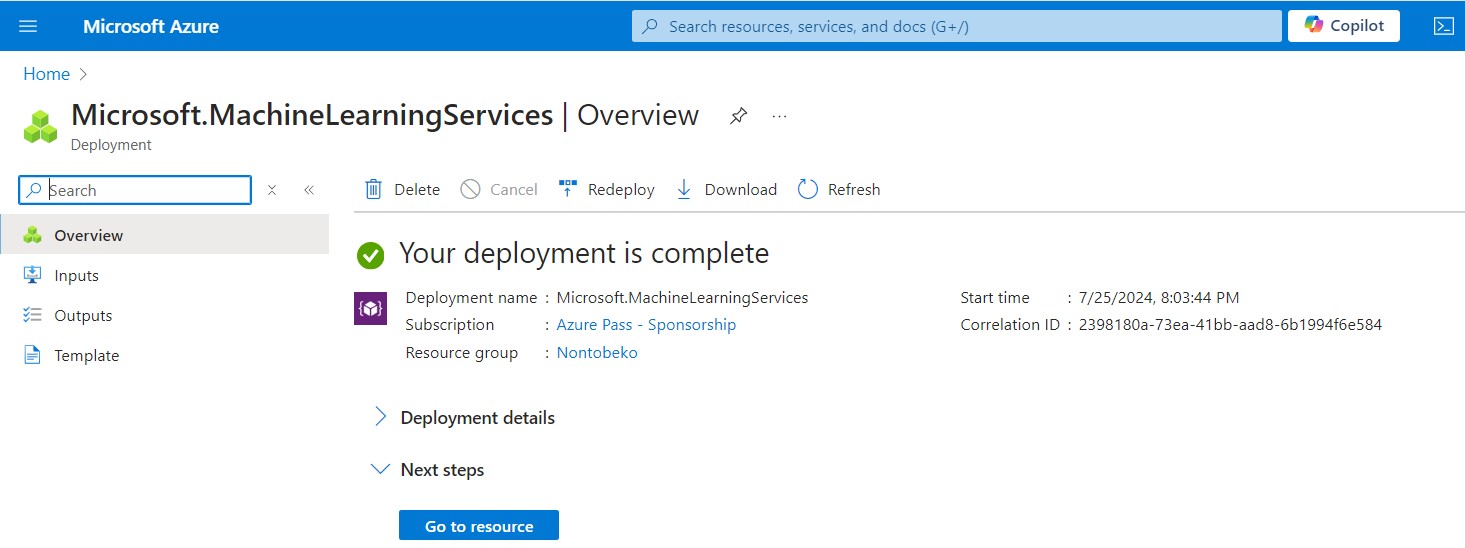
**Software dependencies:** Azure ML SDK, Python 3.8

**Operating System**:I am using Microsoft Windows 11 home Single Language

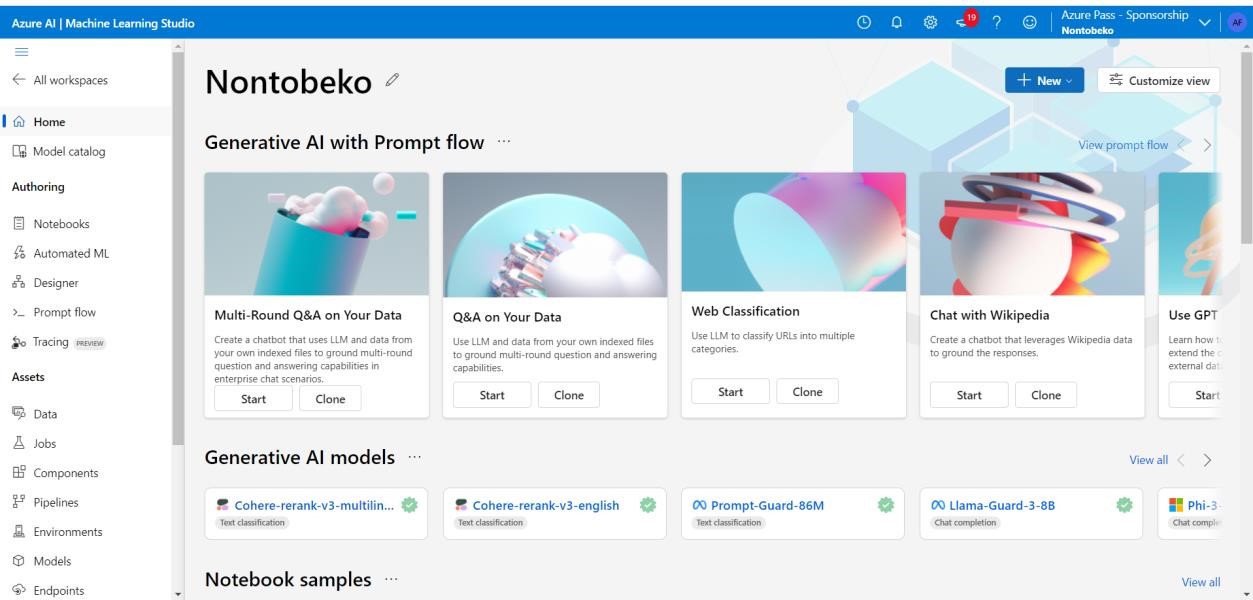
**4. DEPLOYMENT STEPS**

# Step 1 : Create AIML Workspace

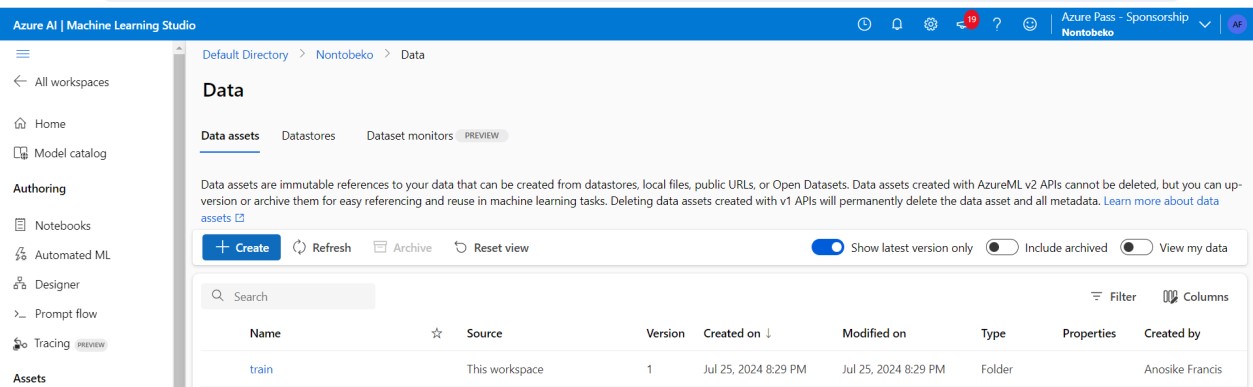
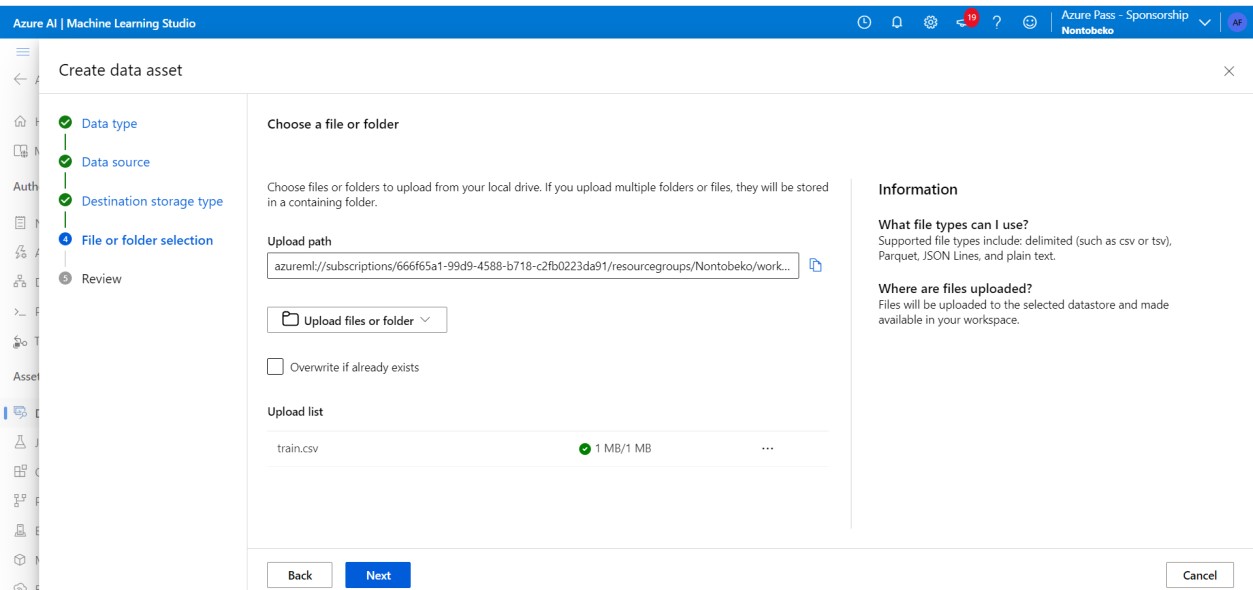




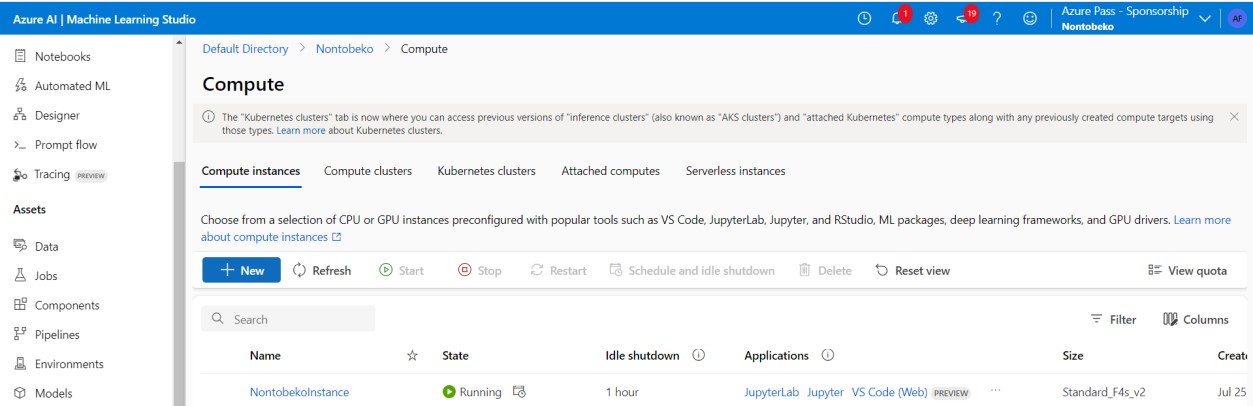
# Step 2 : Open the workspace studio



**Step 3 : Upload the Employees Train dataset under Data component within the workspace.**

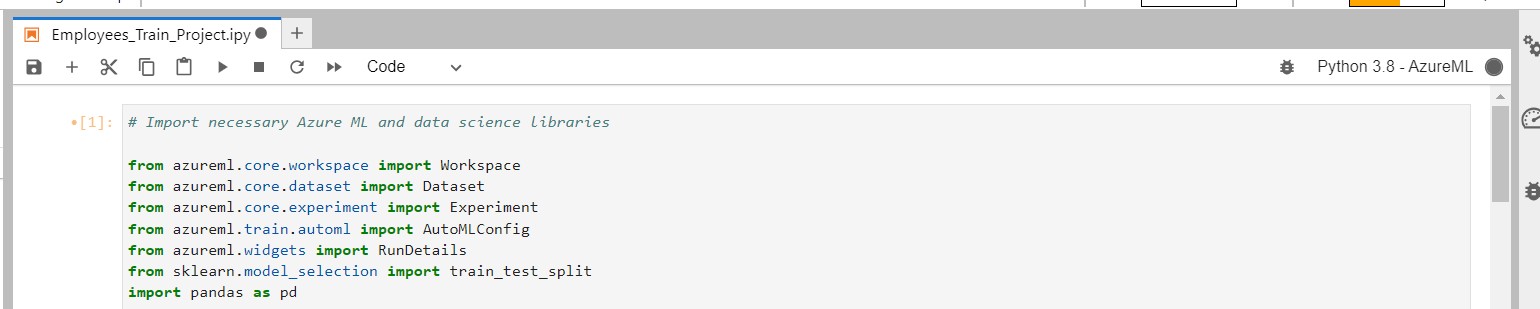
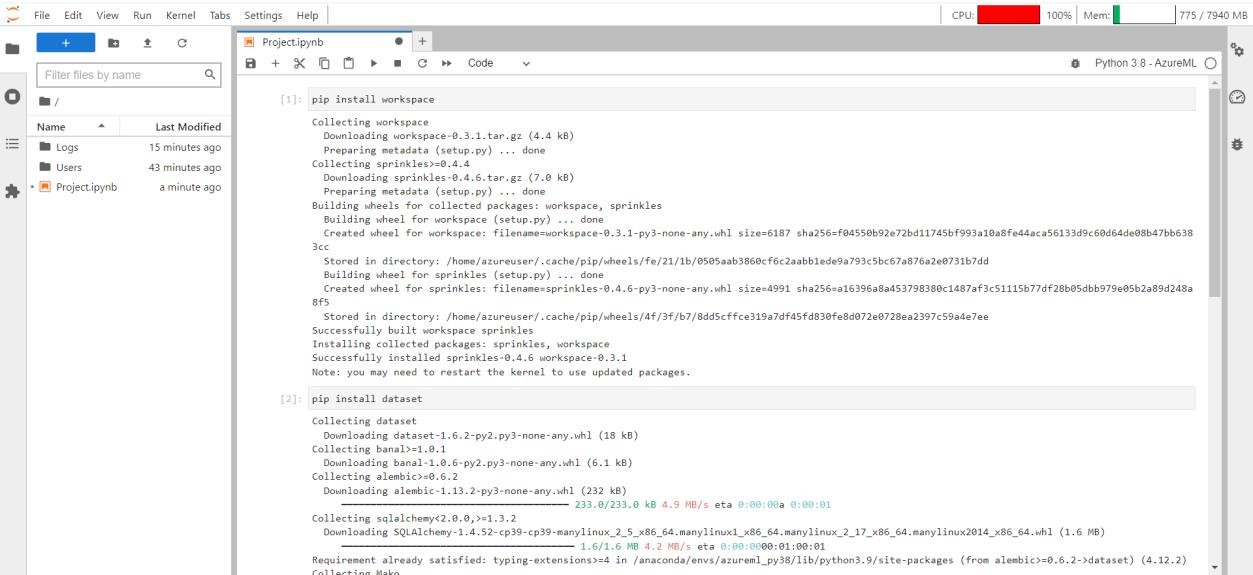


# Step 4: Create the compute instance and open the Jupyter Lab

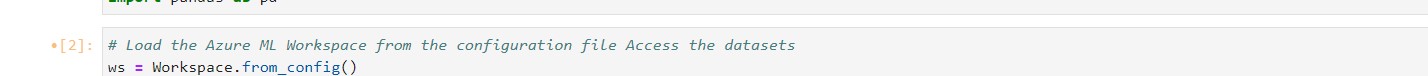


**Step 5: Use the python 3.8 Azure ML**

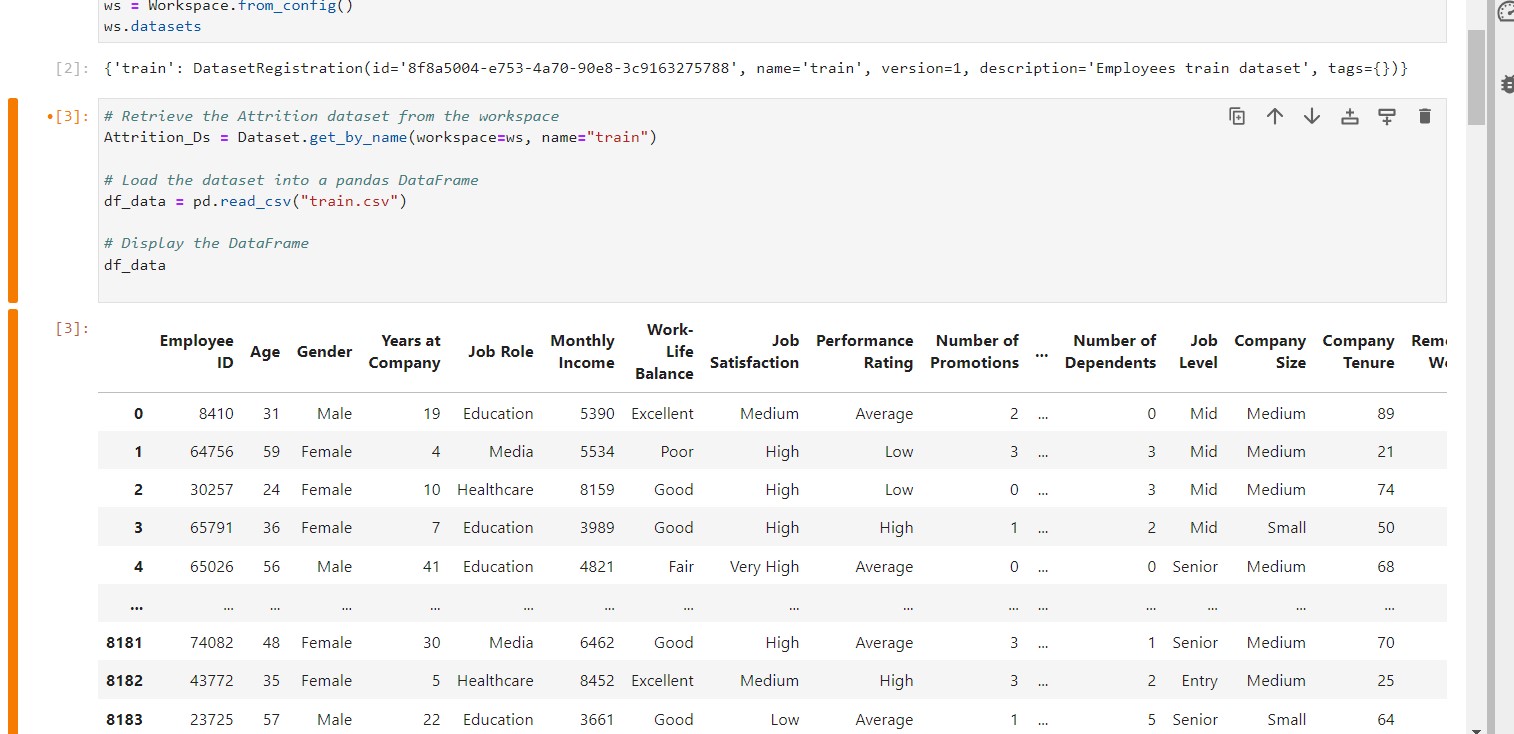
# Import the necessary packages and libraries



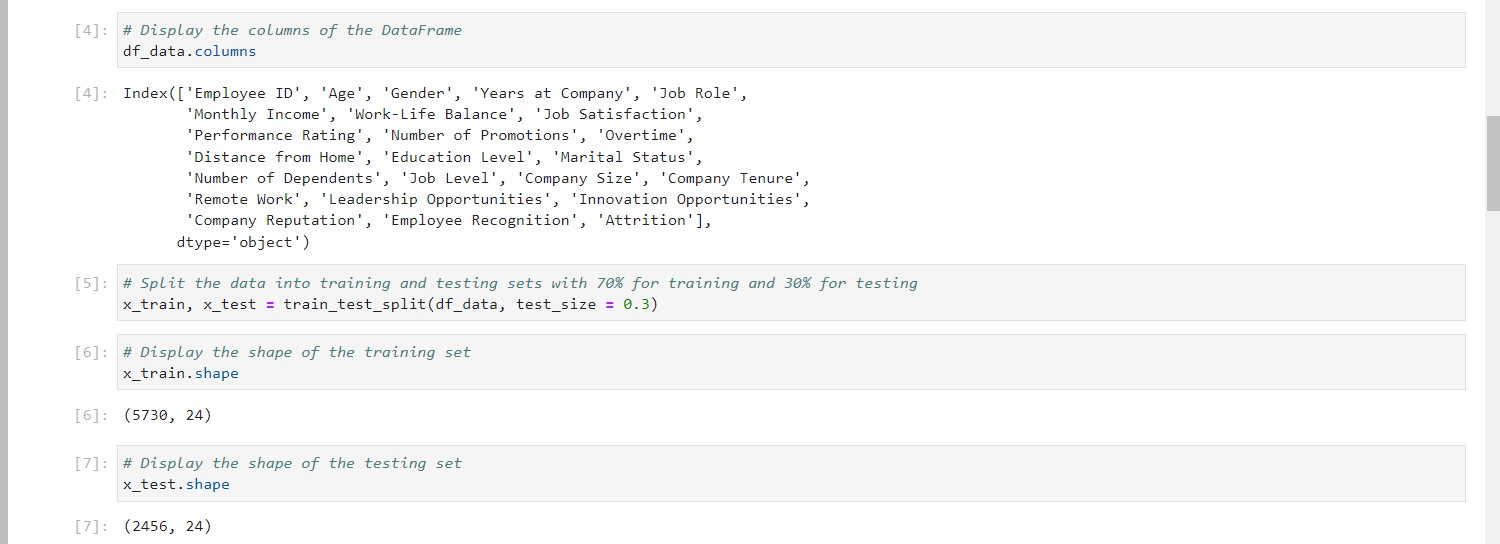
# Step 6: Connect to your workspaces



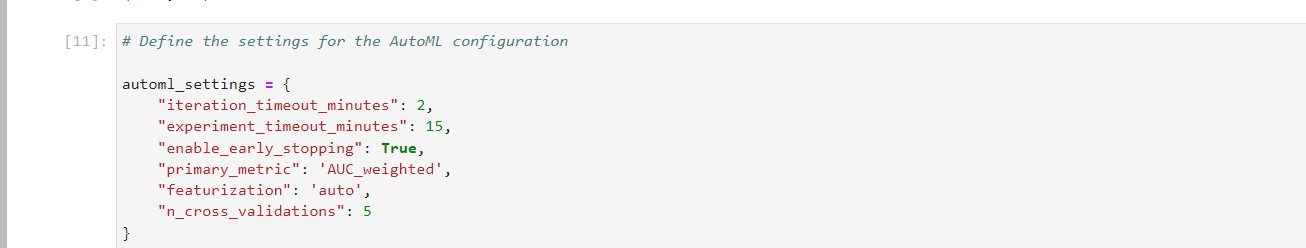
# Step 7: Work with the datasets and read your data. (I used Pandas)



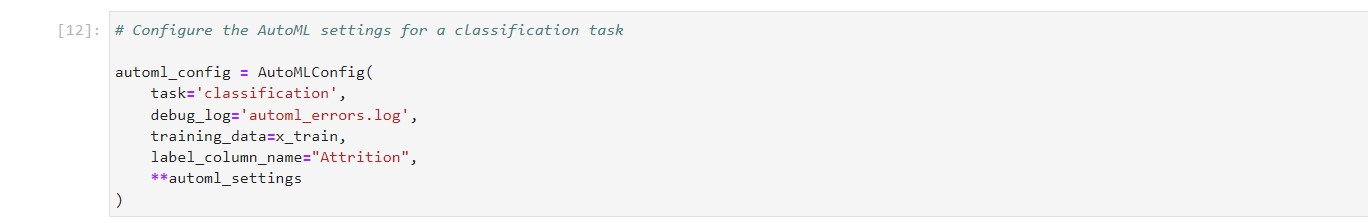
# Step 8 : Train and split your data



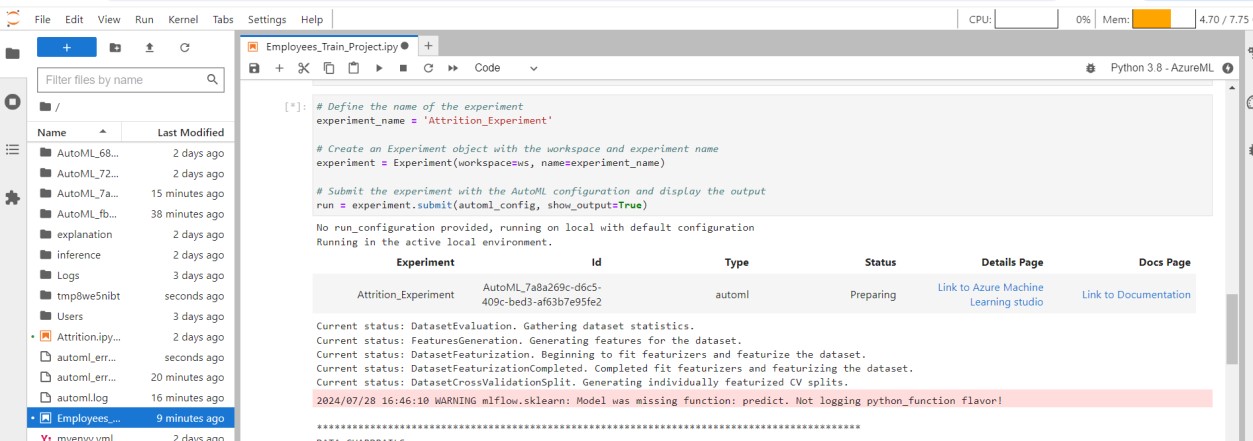
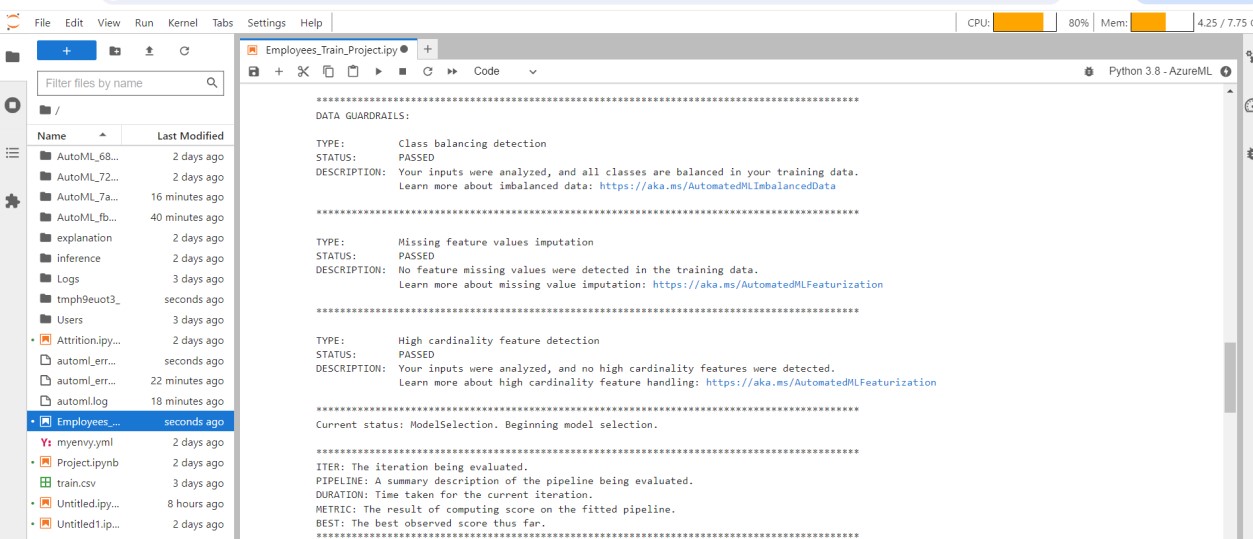
# Step 9: Set up your automl and your experiments settings

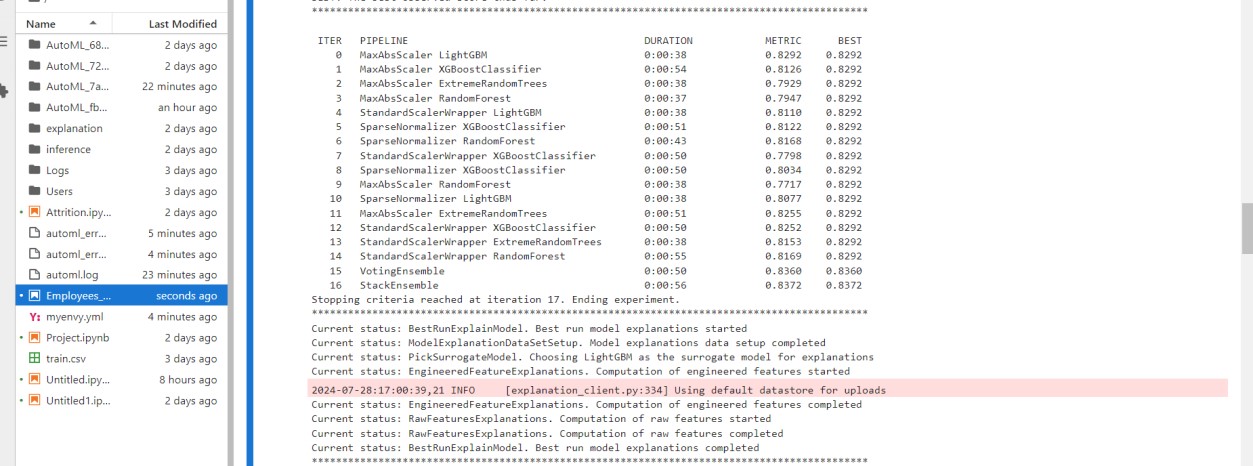


# Step 10: Specify the task and algorithm to use and the specie column as your label (dependent variable)



# Step 11: Create your experiment to use for deployment

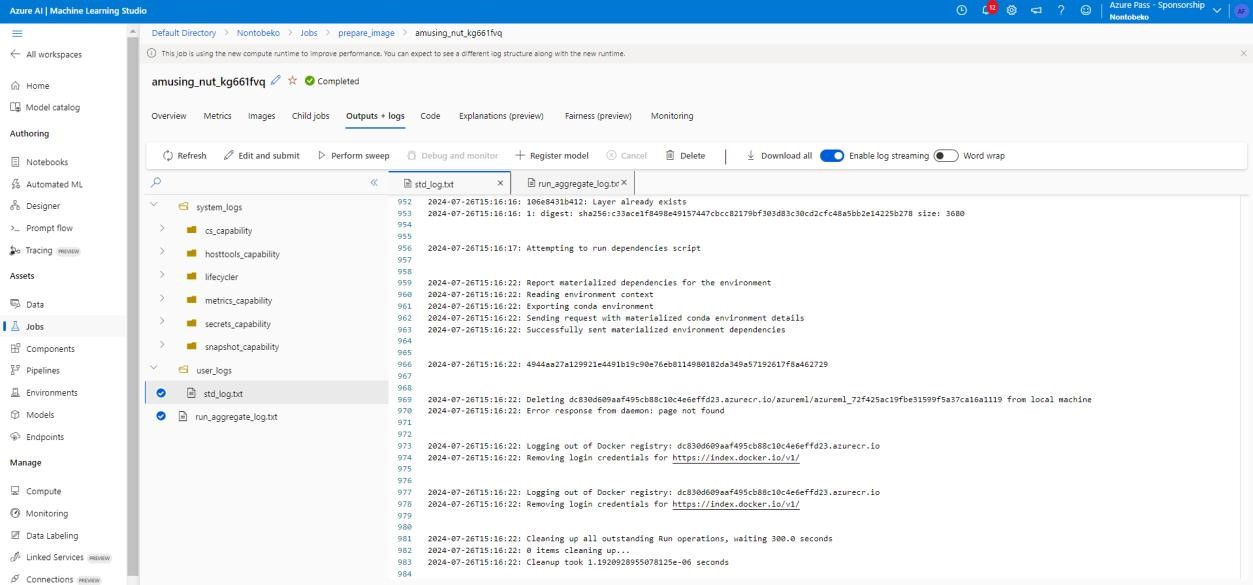
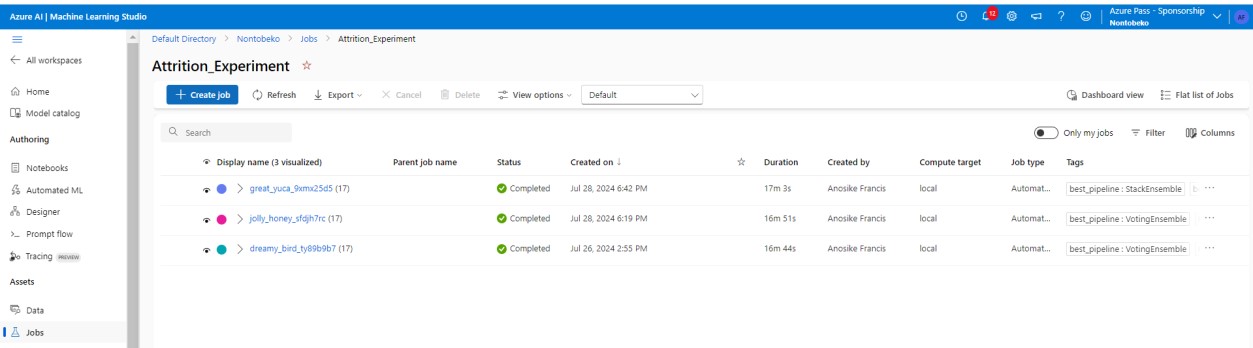




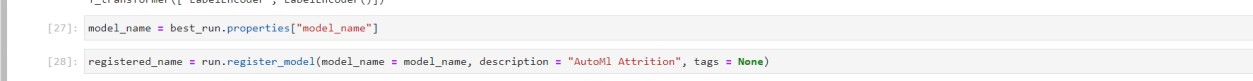
# Step 12: Get the run output



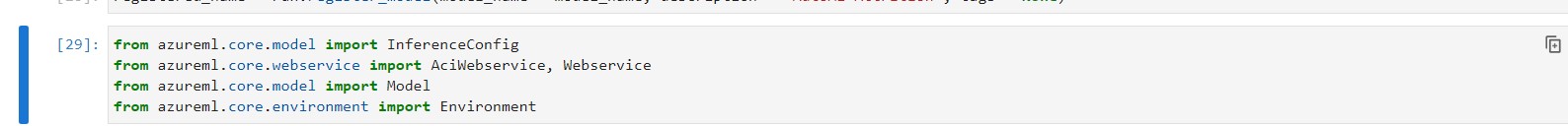
# Step 13: See the experiment and the scoring file created



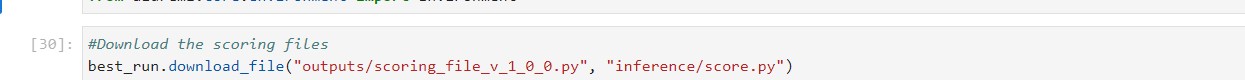
# Step 14: Create and register your model



# Step 15: Import the packages for deployments



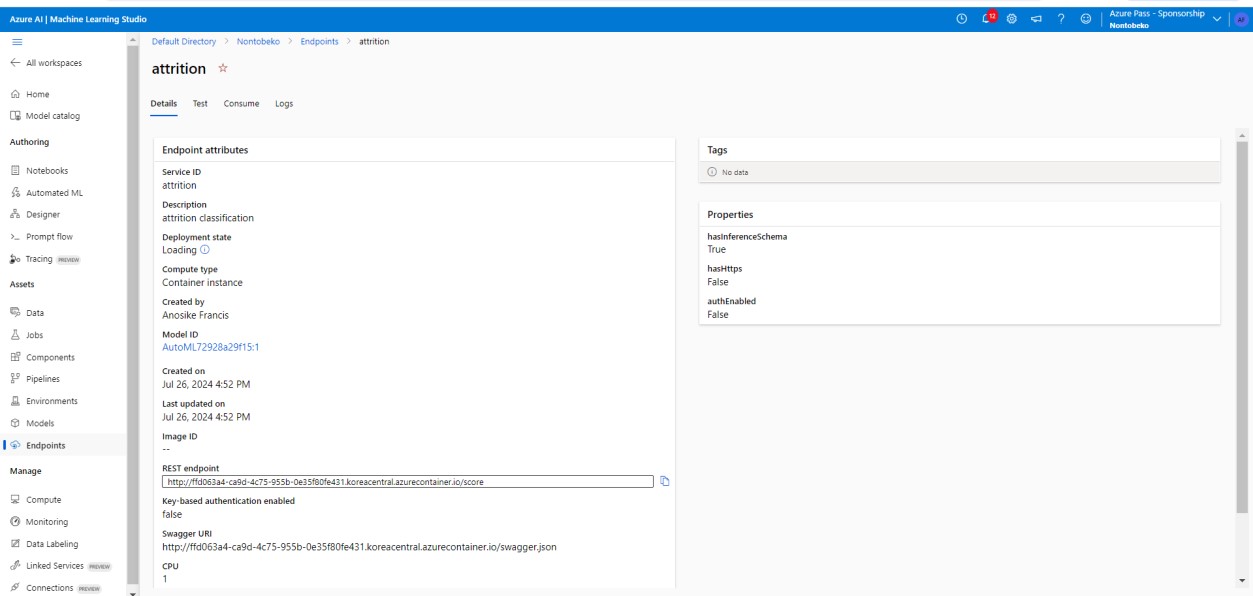
# Step 16: Download and bring in the scored .py file



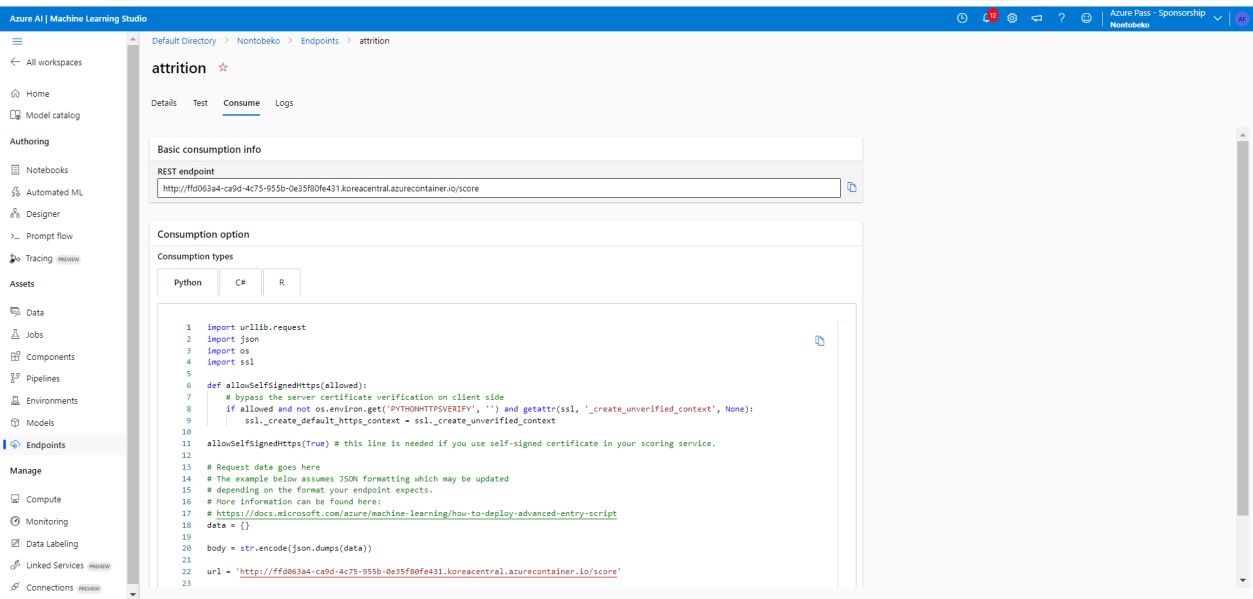
# Step 17: Wait for the deployment to complete (10 – 20mins)



# Step 18: Look at the completed deployment and copy the url link and test it



**Test the predicted result: REST Endpoint link** [**http://ffd063a4-ca9d-4c75-955b-0e35f80fe431.koreacentral.azurecontainer.io/score**](http://ffd063a4-ca9d-4c75-955b-0e35f80fe431.koreacentral.azurecontainer.io/score)



# 5. CONFIGURATION SETTINGS

automl\_settings = {

"iteration\_timeout\_minutes": 2,

"experiment\_timeout\_minutes": 15,

"enable\_early\_stopping": True,

"primary\_metric": 'AUC\_weighted',

"featurization": 'auto',

"n\_cross\_validations": 5

}

automl\_config = AutoMLConfig(

task='classification', debug\_log='automl\_errors.log', training\_data=x\_train, label\_column\_name="Attrition",

\*\*automl\_settings

)

# 6. TESTING AND VALIDATION

Dataset : Train.csv from Kaggle – Employee Attrition

[(https://www.kaggle.com/datasets/stealthtechnologies/employee-attritiondataset?select=train.csv)](https://url.za.m.mimecastprotect.com/s/X5NdCk5j4GTOxq6QS2b2BL?domain=kaggle.com)

Test the predicted result: REST EndPoint [(**http://ffd063a4-ca9d-4c75-955b0e35f80fe431.koreacentral.azurecontainer.io/score)**](http://ffd063a4-ca9d-4c75-955b-0e35f80fe431.koreacentral.azurecontainer.io/score)

Procedure for testing the deployed model to ensure it performs as expected:

1. Environment Configuration
2. Data Preparation
3. Input Data Validation
4. Testing (Use typical examples of data that the model is expected to encounter in production)
5. Prediction Output & Accuracy Assessment
6. Performance Testing (Latency & Throughput
7. Integration Testing
8. Validation Against Baselines
9. Bias and Fairness Testing (if applicable)
10. Documentation of Testing Results
11. Based on testing results, iteratively refine the model if necessary, addressing any identified issues or performance gaps.

# 7. MONITORING AND LOGGING

Monitoring the performance and health of a deployed model is crucial for ensuring it continues to operate effectively and meets service level expectations. Azure provides several tools and services that can be leveraged for performance monitoring.

## Azure Monitor

Metrics - Collects performance metrics such as CPU usage, memory usage, and response times of the deployed model endpoint.

Alerts - Set up alerts based on predefined thresholds for metrics, for example if response time exceeds a certain limit.

Logs: Azure Monitor can also collect logs from various Azure services, including Application Insights and Azure Machine Learning, to provide deeper insights into model performance.

Logging mechanisms are essential for troubleshooting, debugging, and auditing purposes.

## Azure Monitor Logs

Querying Logs - Use Azure Monitor Logs to query and analyse logs collected from various Azure services, including Application Insights and Azure Machine Learning.

Log Analytics - Leverage Log Analytics to perform advanced queries, create dashboards, and gain insights into the operational health of the deployed model.

# SCALABILITY AND PERFORMANCE

## Scalability Considerations

When scaling a machine learning model to handle increased traffic or larger datasets on Azure, several considerations come into play to ensure optimal performance and efficiency:

▪ Compute Resources, Auto-scaling, Data Storage, Load Balancing, and Caching.

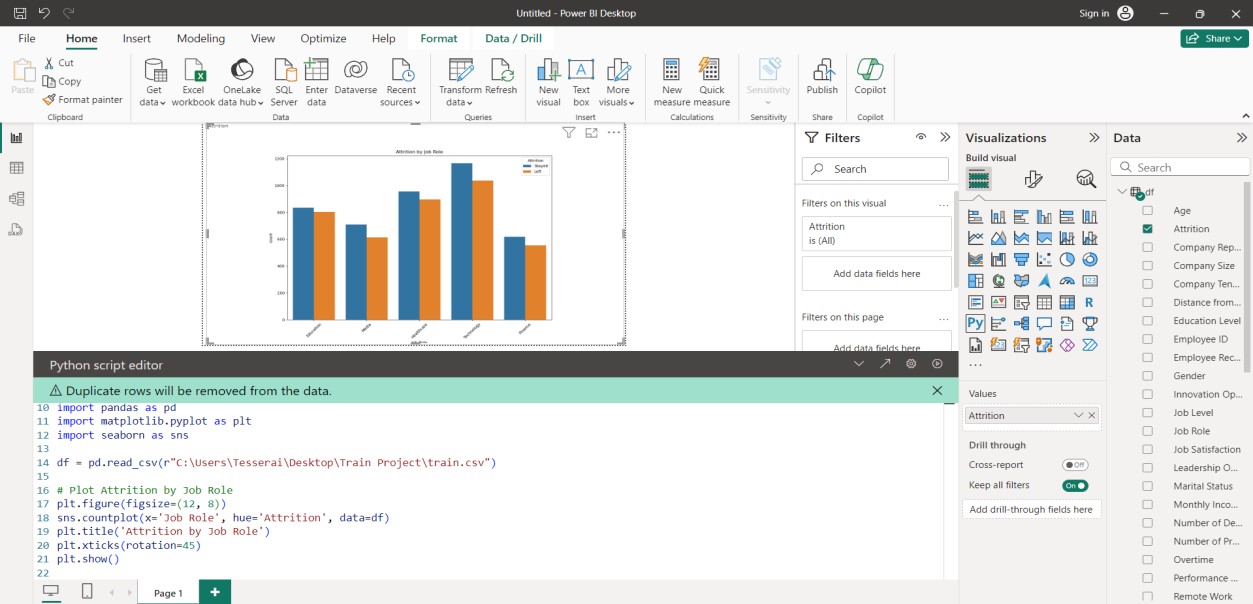
## Performance Optimization

Optimizing the performance of a machine learning model on Azure involves enhancing its speed, efficiency, and resource utilization. Here are techniques and benchmarks for achieving optimal performance:

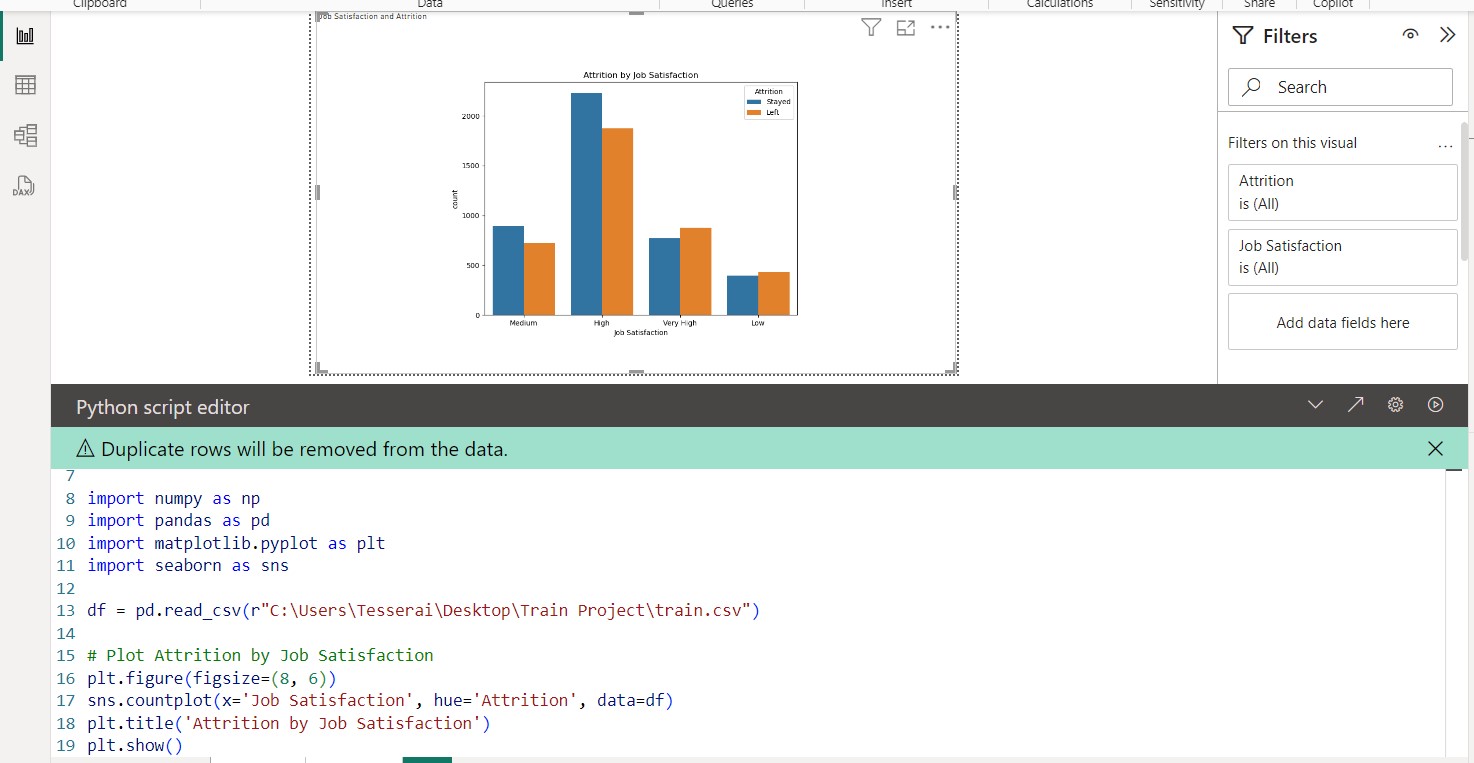
▪ Model Optimization, Hardware Acceleration, Batch Processing, Model Compression, Pipeline Optimization, Benchmarking and Monitoring.

# DATA VISUALIZATION USING POWER BI

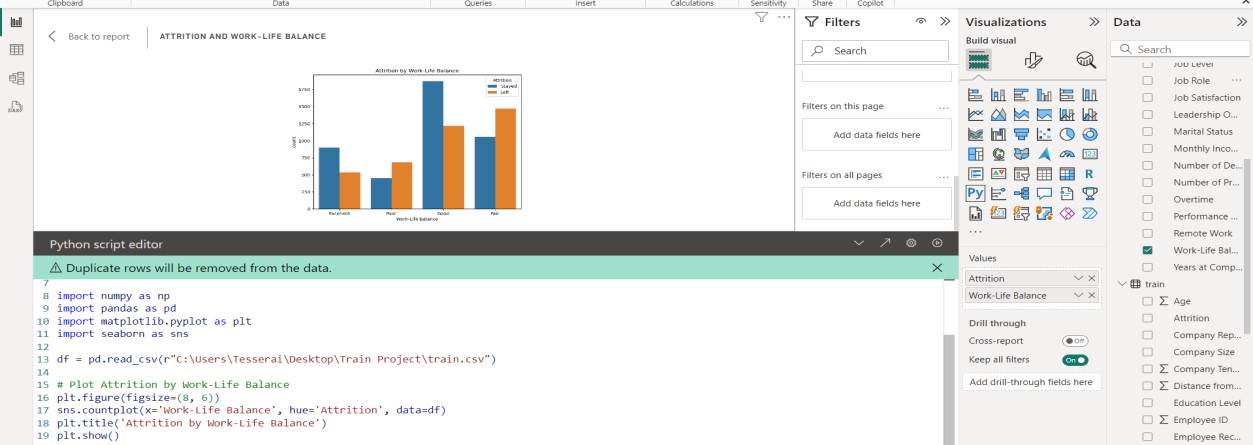
## Plot Attrition by Job Role



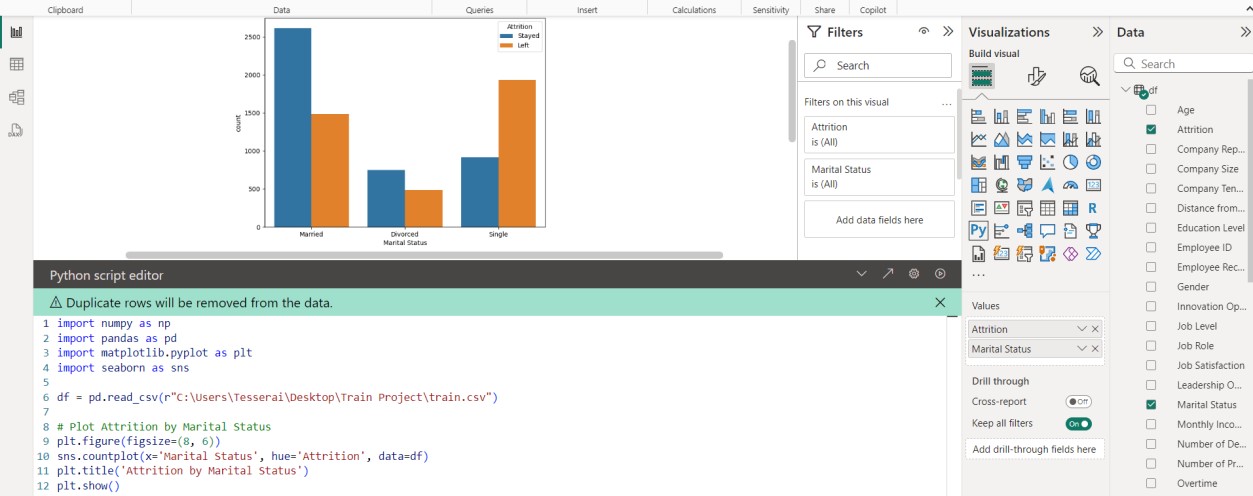
## Plot Attrition by Job Satisfaction



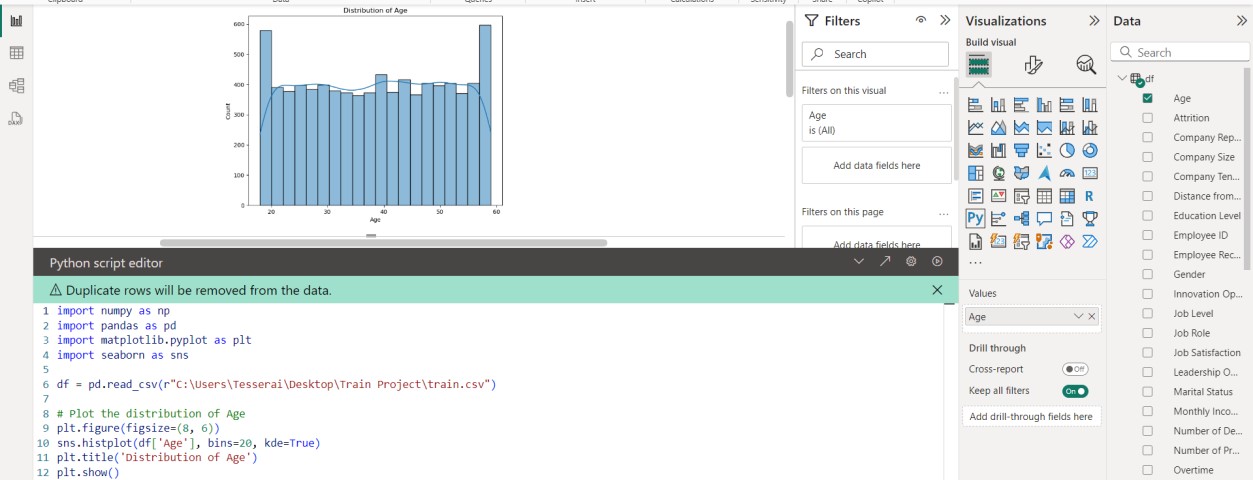
## Plot Attrition by Work-Life Balance



## Plot Attrition by Marital Status

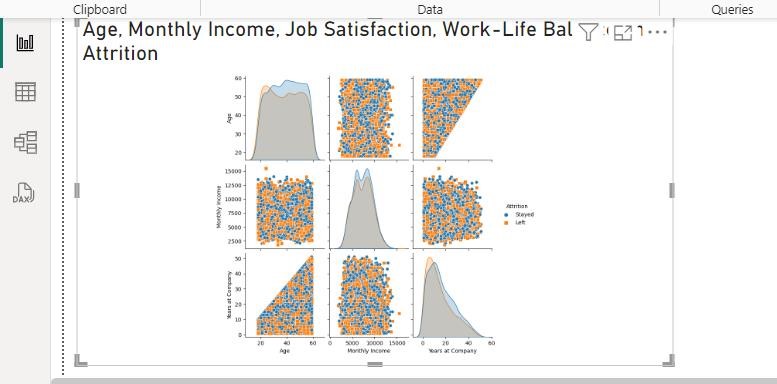


## Plot the distribution of Age

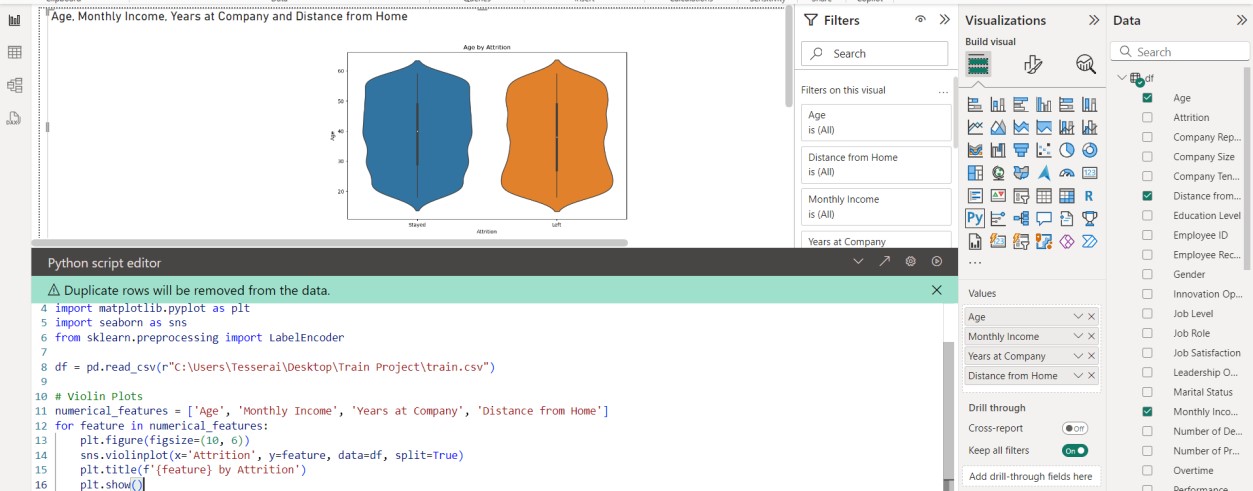


## Plot the confusion matrix



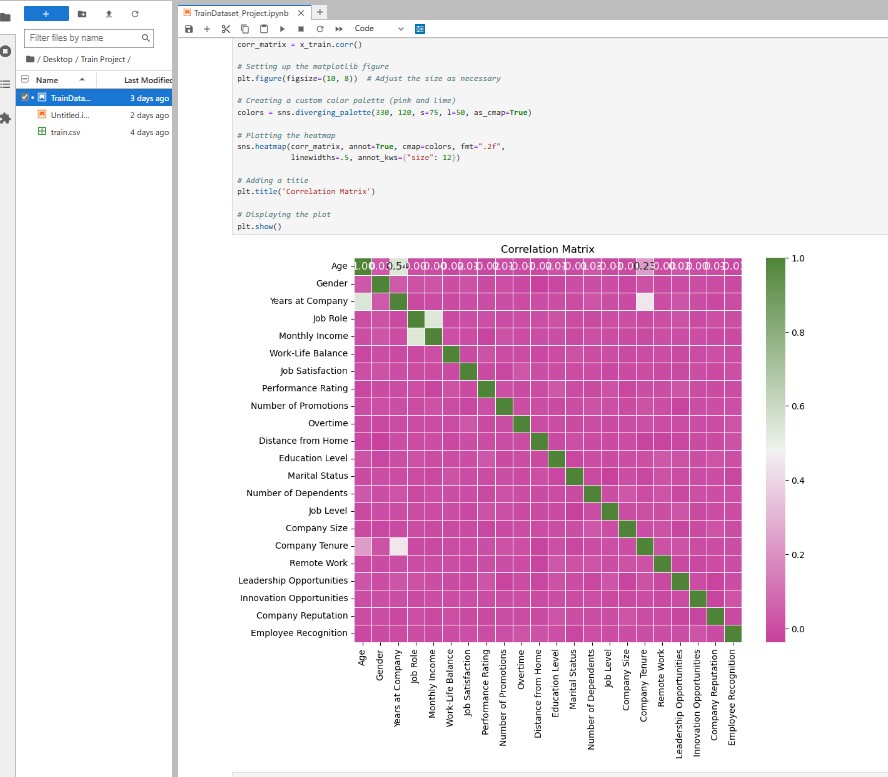


## Violin Plots



# DATA VISUALIZATION USING PYTHON

## Plotting Hitmap



## Plotting the Data Distribution

