# Part One: Classification Algorithm

# (Python and Azure Machine Learning Designer)

# 1. Title

## a. Classification Algorithm: Python

The aim of this task was to predict the income level of adults across a selected demographic. The title of this task is “Correlative Analysis of Classification Algorithm in Predicting Income Levels using Python”

## b. Classification Algorithm: Azure Machine Learning Designer

This task has a semblance to that described above, with the same aim. The title of this task is “Comparative

Income Level Classification: Leveraging Azure Machine Learning Designer”

# 2. Introduction

This analytical research explores the area of predictive modelling using the Adult Dataset. The main objective of this analysis is to use classification algorithms to predict income levels of a selected demographic.

This dataset offers an interesting demographic containing rich attributes making it one of the bestpick candidates for classification problems.(Chakrabarty & Biswas, 2018)

The dataset carries a broad range of variables including race, gender, marital status, occupation, etc., These attributes come together and give analytical balance to our analysis- indicating whether an individual’s annual income is above $50,000 ($50k). The binary nature of the attribute of interest makes this dataset a clear choice.

This task would be performed utilizing both Python (leveraging on its dynamic robustness and processing capabilities)(Rayhan & Kinzler, 2023) and the Azure MLD (leveraging on its cloud-based platform built for machine learning models)

In applying these tools, we not only gain an analytical understanding of the income level determinants but also gain more context from a socio-economic point of view. This may offer broader insights into the dynamics of income distribution amongst similar demographics.

# 3. Dataset Description

The principal focus for this analysis is the Adult Data Set from [UCI Machine Learning Repo.](https://archive.ics.uci.edu/dataset/2/adult)

The dataset is comprised of 32,561 instances (excluding column names) and 15 features (including the target feature).

The target feature in this dataset is “income” which can be classified as binary classifying weather or not the income is exceed $50k or not, i.e ('>$50k' )( '≤$50k' ) making this dataset ideal for this kind of classification task.

In addition, the other attributes are not existing in isolation. They are in association, hence offering a comprehensive overview of the data-story telling. This is a great characteristic for a dataset with potential to make real-work relevance.

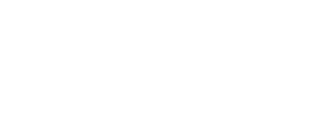
|  |  |
| --- | --- |
| **Variable** | **Variable Type** |
| Age | Integer |
| Workclass | Categorical |
| Fnlwgt | Integer |
| Education | Categorical |
| Education Num | Integer |
| Marital Status | Categorical |
| Occupation | Categorical |
| Relationship | Categorical |
| Race | Categorical |
| Sex | Binary |
| Capital-Gain | Integer |
| Capital-Loss | Integer |
| Hours-Per-Week | Integer |
| Native-Country | Categorical |
| Income | Binary |

*Table 1: Variable Information of the adult dataset from UCI.* (Ronny Kohavi & Barry Becker, 1996)

# 4. Data Exploration and Preprocessing

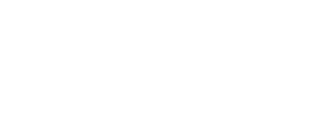
The data processing step for this analysis is a critical step to ensure data quality suitable for accuracy in our model training and in turn the evaluation.

The steps used here are cartegorized into three (3) as illustrated below:

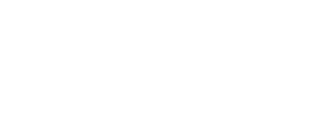


Handling missing

values



Outlier Detection

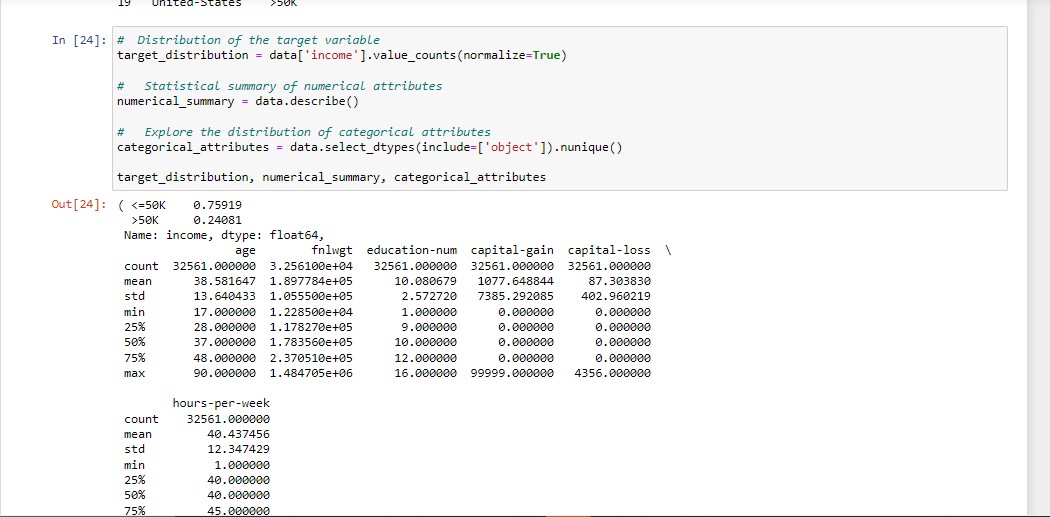
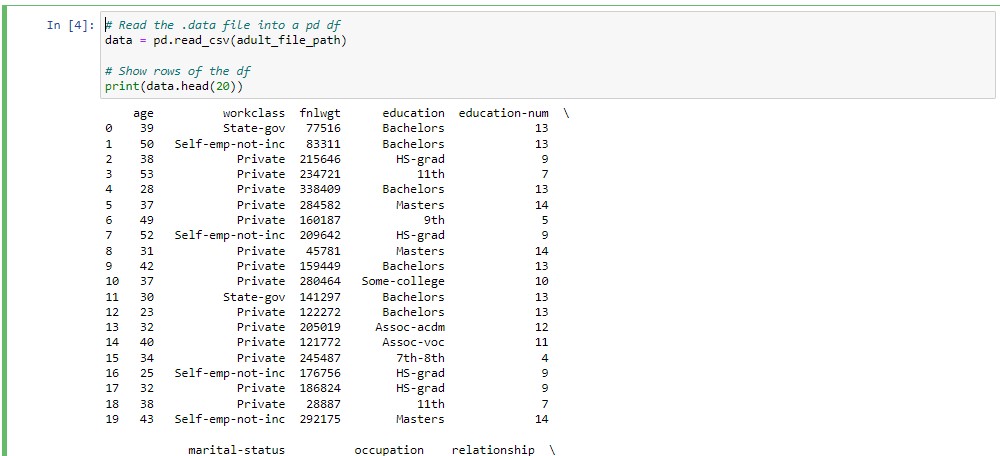
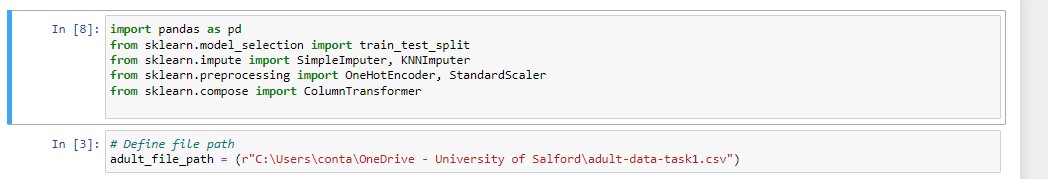


Data transformation

& Encoding

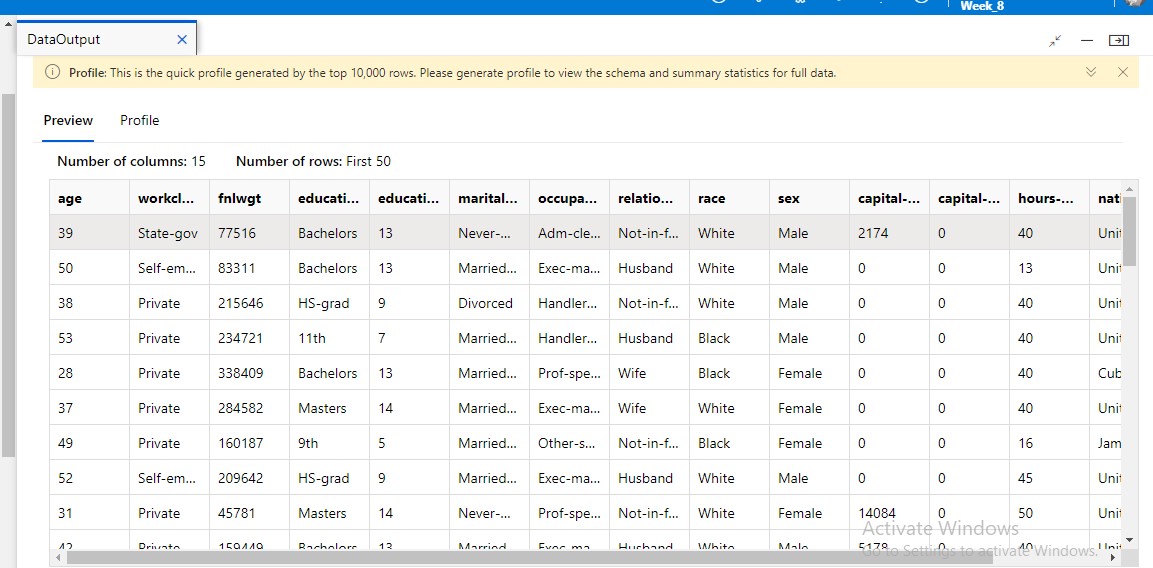
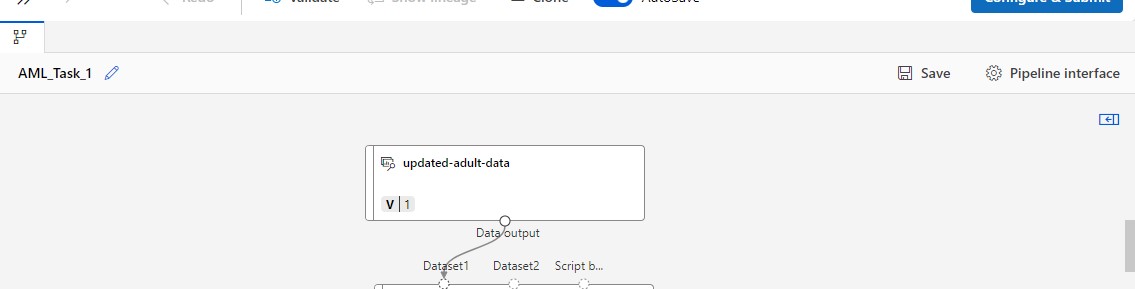
## i. Import necessary libraries, Load Data and Inspect the Data

* **PYTHON:**

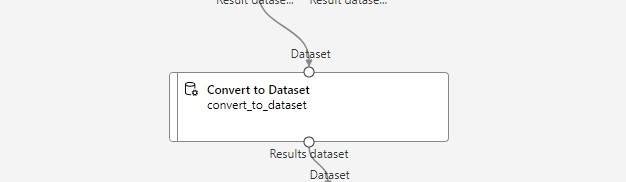


* **AML:**

The Adult-dataset has been uploaded from local machine and the output previewed to confirm correct schema and successful file upload.



-Convert to ‘Dataset’: This step is included in the workflow to ensure that the data is in a suitable format for ML models in Azure. In addition to providing suitability to the data for any required manipulation.

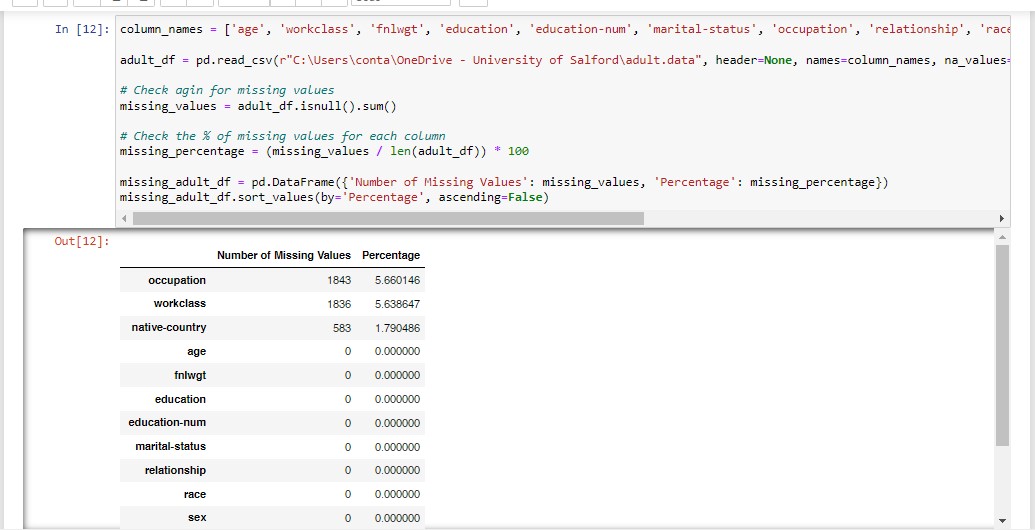


## ii. Handling Missing Values

• **PYTHON:**

Upon physical examination of the dataset, missing values were spotted hidden as question marks “?’. Two different methods were used to resolve this. These methods were chosen with the strong aim to preserve the original distribution of the variables whilst nothing the small percentage of values missing.

1. Mode Imputation for ‘Workclass’ and “Occupation”: These variable exists as categorical varaiables exhibiting strong mode. Hence the use of mode imputation
2. KNN Imputation for “Native-Country”-This variable was more diverse in nature and a this method was chosen to be suitable for variables with complex relationship with others.



The summary of missing values:

-Occupation: 1,843 missing values. Approximately 5.66% of data.

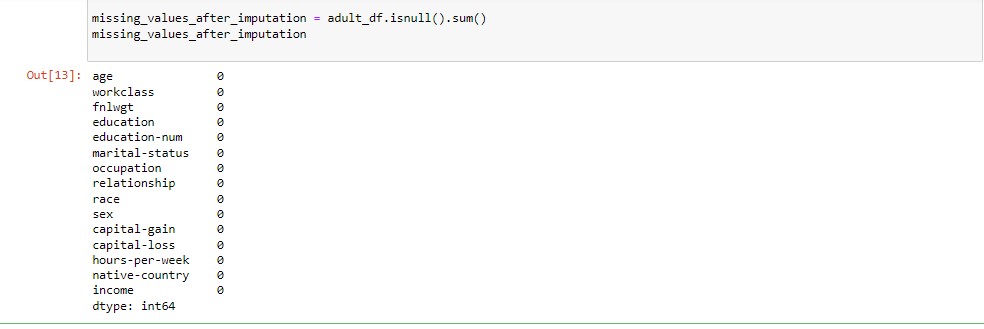
-Workclass: 1,836 missing values Approximately about 5.64%.

-Native-country: 583 missing values, around 1.79%.

These missing values were initially hidden as question marks and are now correctly identified as NaN in our Data frame.

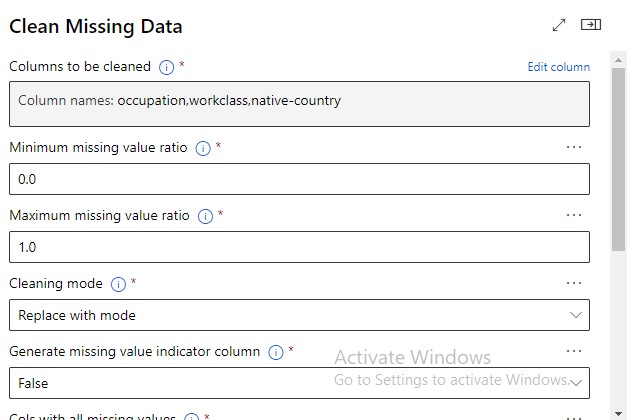
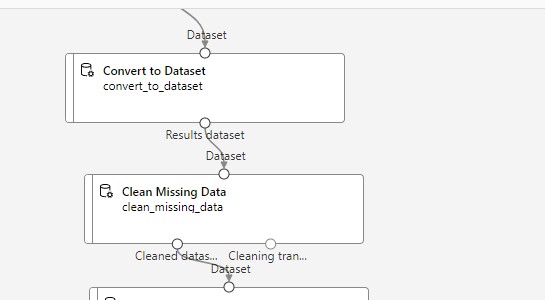


After the missing value handling Techniques, there are no more mssing values within the dataset.



• **AML:**

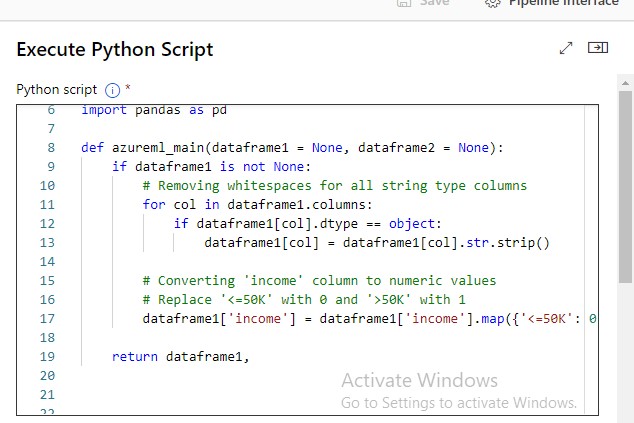
The same variables with missing values as in with the python method. To tackle this, the “Clean missing data” module was employed in the next step within the work flow.



The missing data was cleaned using the “Mode Imputation” method.

## • Executing python scripts

Azure MLS is detecting the target variable as a categorical attribute instead of a binary attribute. To resolve this, the “Execute Python Script” module has been employed to modify the variable.



### iii. Detecting Outliers

Outlier detection was conduction numerical attributes using the Interquartile Range method, chosen due to the peculiarity of our data

A total of 13,564 outliers was discovered which is a substantial figure. Suggesting that removing these data points would lead to significant loss of important information.



Determining the best management methods would have to be analytically handled since they could potentially impact the model training and predictions. Taking domain knowledge and the distribution of the individual variables, two methods would be employed:

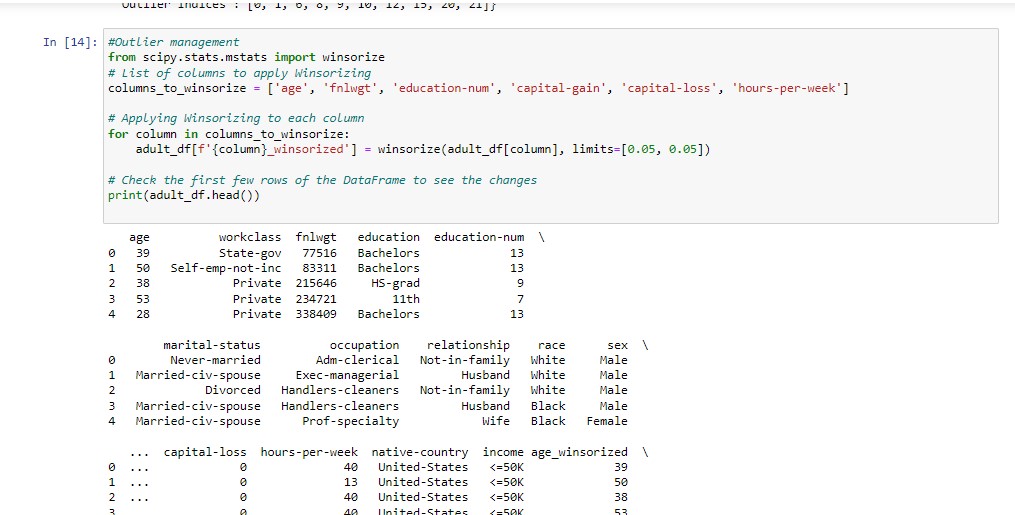
1. Winsorizing the numerical columns- involving limited the extreme values in data distribution
2. Employing robust modelling methods that are less sensitive to outliers.

iv.

Data

Encoding

and Data Normalization



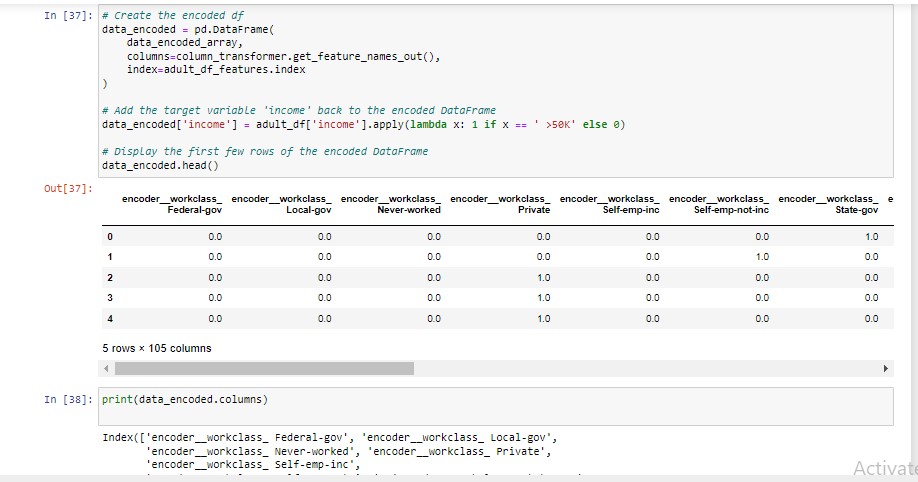
• **PYTHON:**

## Data Encoding

This is usually performed to prepare the dataset of ML algorithms. One Hot Encoding for Categorical Variables: Since the ML (machine learning) models will require numerical input. Categorical variables within the dataset were transformed into a set of binary variables.



Encoding completed.

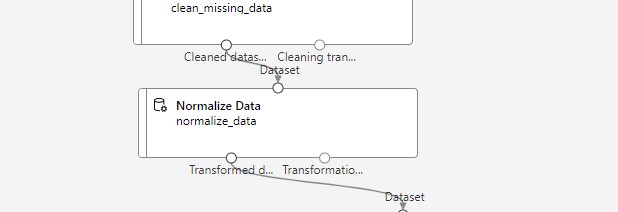
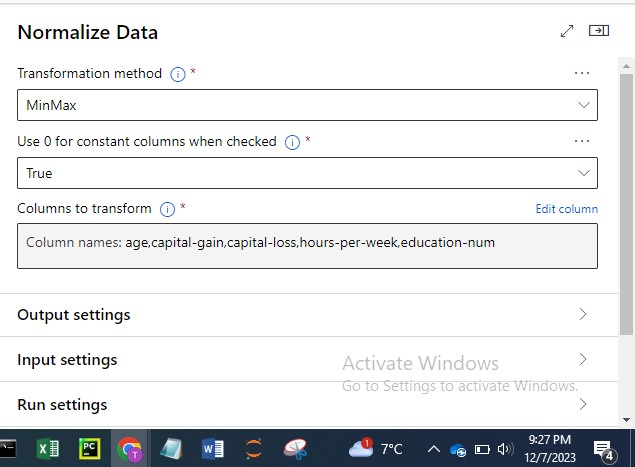


• **AML:**

## Data Normalization

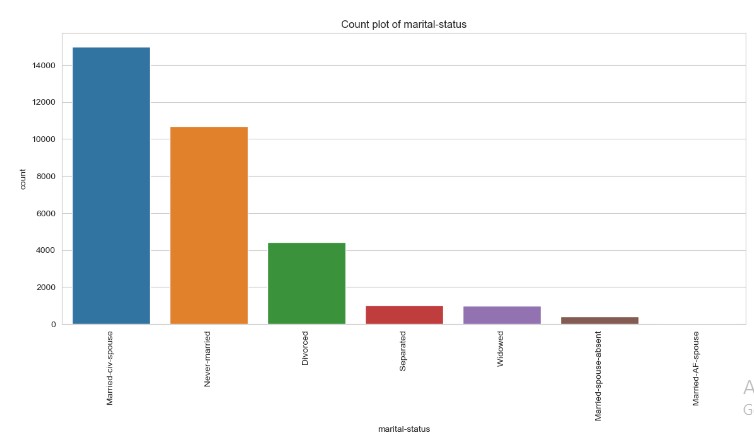
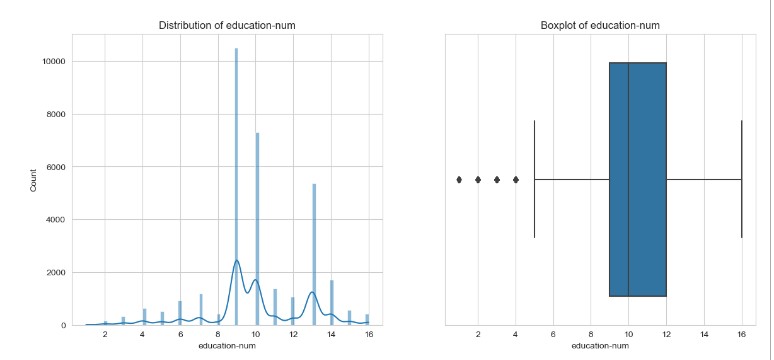
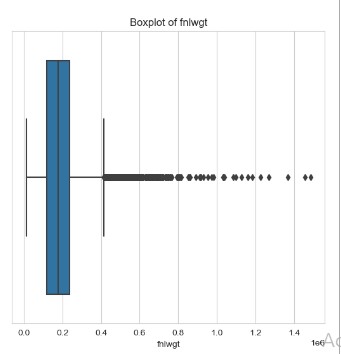
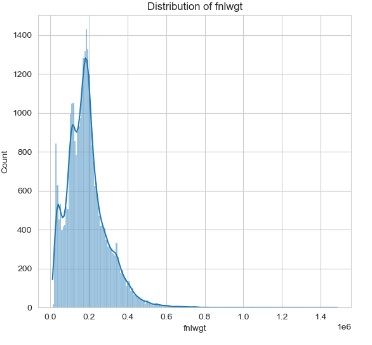
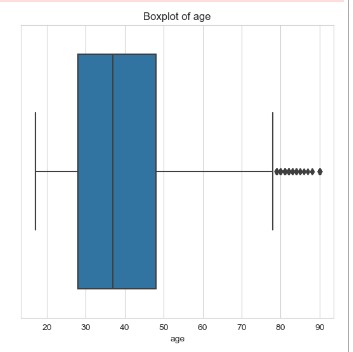
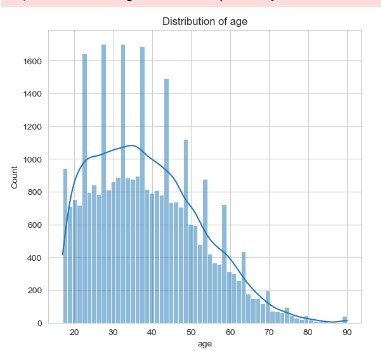
Normalization is a crucial step in data pre-processing when working in Azure Machine Learning studio for classification problems. Normalization adjusts the scale of different features to a related scale range, usually [0, 1]. This ensures that no feature dominates the model due to its scale.

In addition, algorithms (e.g. KNN,SVM) that depend on distance perform better when data is normalized.



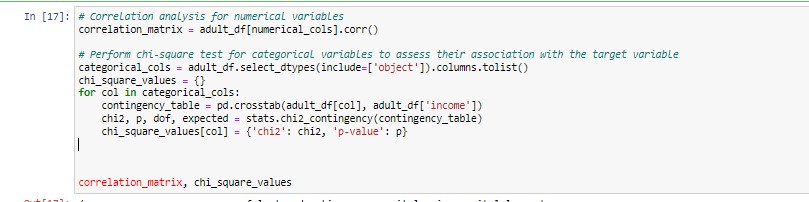
### v. Visual Exploration

These self-explanatory plots highlight the interaction and distribution of variables. Allowing a wideeyed view using different kinds of plots. Some plots highlighted here are, the distribution of gender, age, marital status, education, etc., amongst the population. More plots are within the code sheet.



### vi. Further Dataset Exploration using Chi-Square and Correlation Matrix Score

To further dive deep to explore the dataset, the Chi-square and Correlation matrix were employed (for testing independence between categorical variables and target variables and for testing linear relationships between numerical variables respectively)



a. Correlation Results:

-‘Age’ shows a modest positive correlation with ‘education-num’ 0.036527 and ‘capital-gain’ 0.077674, inferring that older individuals may perhaps have higher education and more capital gains. -‘Education-num’ correlates positively with hours-per-week 0.148123, suggesting that highereducated individuals tend to work more hours.

-‘Capital-gain’ correlates positively with age and ‘education-num’, indicating to a linkage between age, education, and investment income.

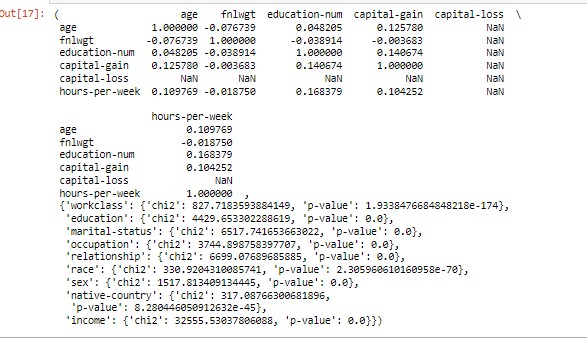
(Kumar & Chong, 2018)

Chi-Square Test Results:

-‘Income level’ is seen to have strong association/correlation with variables like ‘workclass’ (chi2: 827.72, p-value: ~0.0), education’ (chi2: 4429.65, p-value: 0.0), ‘marital-status’ (chi2: 6517.74, pvalue: 0.0), occupation’ (chi2: 3744.90, p-value: 0.0), ‘relationship’ (chi2: 6699.08, p-value: 0.0),

‘race’ (chi2: 330.92, p-value: ~0.0), ‘sex’ (chi2: 1517.81, p-value: 0.0), and ‘native-country’ (chi2:

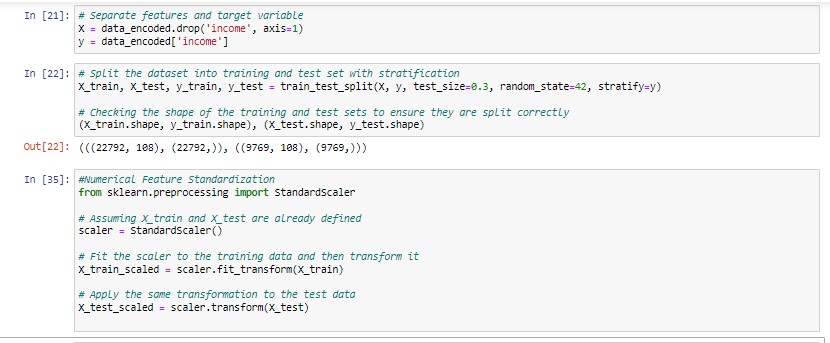
317.09, p-value: ~0.0)



### vii. Numerical Feature Standardization

Standardization was applied to the numerical features in the dataset using the StandardScaler from sklearn.

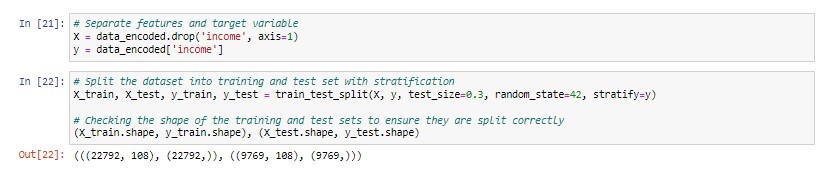
This is such an important step as it normalizes these features by removing the mean and the scaling to unit variance. This is essential for algos that are sensitive to the scale of data.



### viii. Splitting the Data

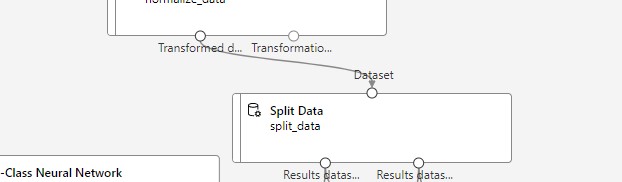
* **PYTHON:**

This process involves splitting the data into appropriate training and testing while maintaining class distribution representation.

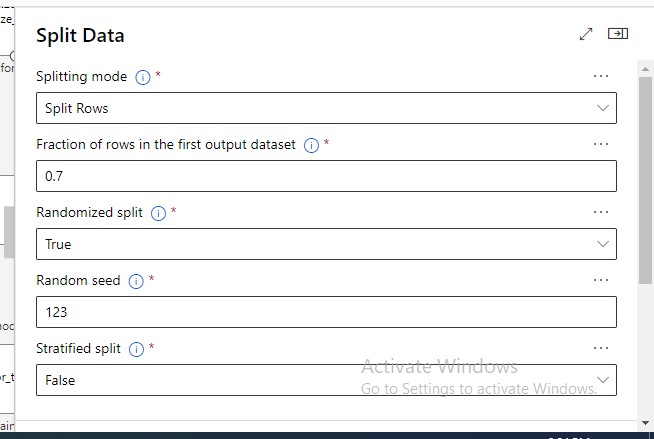


* **AML:**

To split the data for modelling. The ‘split’ data module will be dragged and dropped into the work flow. This step will split the set into “Test” and Training”.



The row split of 0.7 in this data processing step typically means that 70% of the dataset is allocated for training and the rest 30% for testing. This split ratio is chosen to provide a sufficient amount of data for training the model that is needed to learn the underlying patterns, whilst reserving enough distinct data for testing that will be used to evaluate the model's performance on unseen data.

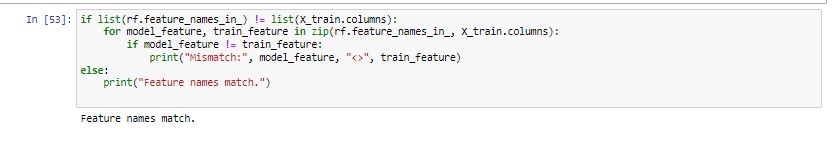
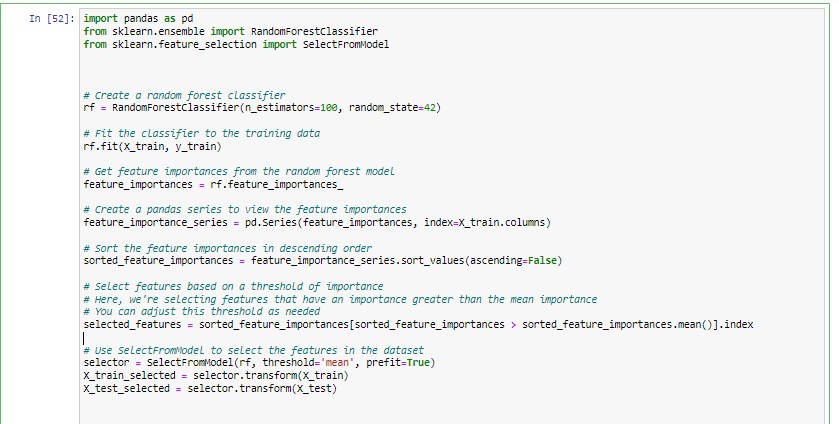


### ix. Feature Implementation

This step details the implementation of feature selection using Random Forest Classifier.

It effectively reduced the number of features in the model on the basis on threshold importance.

Potentially allowing the model to focus on the most relevant predictors and in turn reduce overfitting.



# 5. Model Implementation and Evaluation

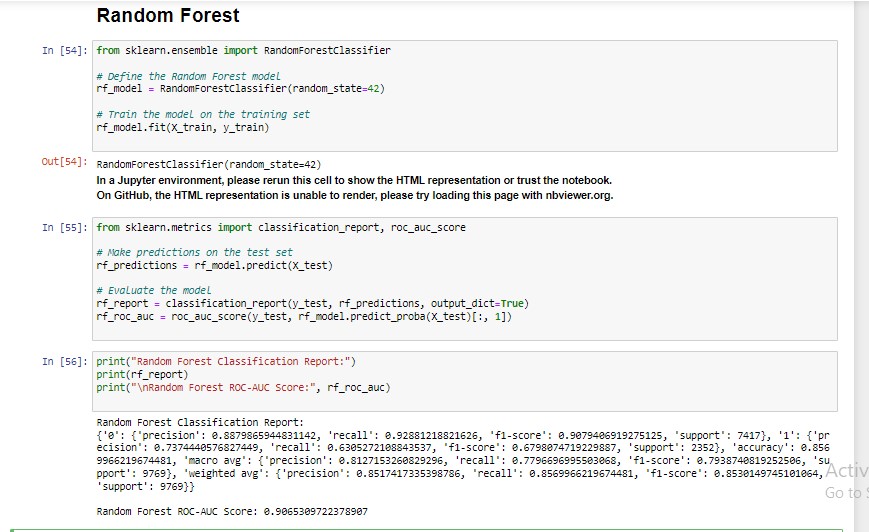
• **PYTHON:**

To perform the required classification analysis, three classification algorithms will be applied and the performance evaluated.

|  |  |
| --- | --- |
| **Algorithm** | **Selection Justification** |
| Random Forest | Robustness, handles varying data types, better accuracy, feature importance |
| KNN | Simplicity and intuitiveness, flexibility in choosing 'k' |
| Decision Tree | Ease of interpretation and visualization, handling non-linear  Relationships, less need for feature scaling |

## i. Random Forest

A Random Forest classifier is here defined with a fixed random state (random\_state=42) for reproducibility reasons.



## ia. Random Forest Results

• Classification Report and ROC-AUC Score:

* The model shows good performing score with an accuracy of approximately 85.7%.

-This model is better at predicting the majority class as seen in the precision for class 0 (income ≤

$50k) is higher than for class 1

-The model can actually identify most of the actual class 0 instances. The high recall for class 0 is high( 0.928)

-The f1-scores, which balance precision and recall, are fairly reasonable, even though better for class 0.

* The model does well at ranking prediction The ROC-AUC score of 0.9065 which is high suggests

this.

* + Accuracy, Precision, Recall, F1 Score, and Confusion Matrix Results

-Accuracy (0.86): This indicates that the model will correctly predict the outcome 86% of the time.

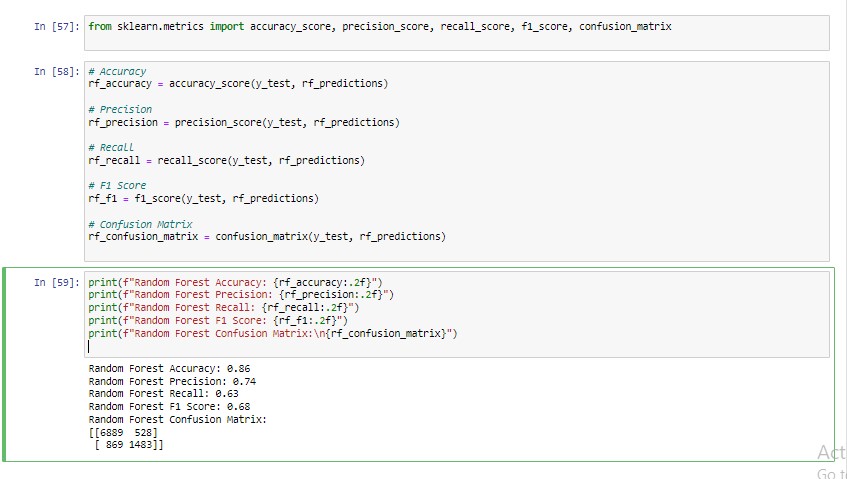
-Precision (0.74): This indicates that whenever the model predicts the positive class (income >$50k), it is correct 74% of the time.

-Recall (0.63): This shows that the model identifies 63% of all actual positive cases.

-F1 Score (0.68): The F1 score is the symmetric mean of precision and recall. A higher F1 score, such as this indicates a pretty good balance between precision and recall, and a score of 0.68 suggests a relatively good balance in this case.

-Confusion Matrix:

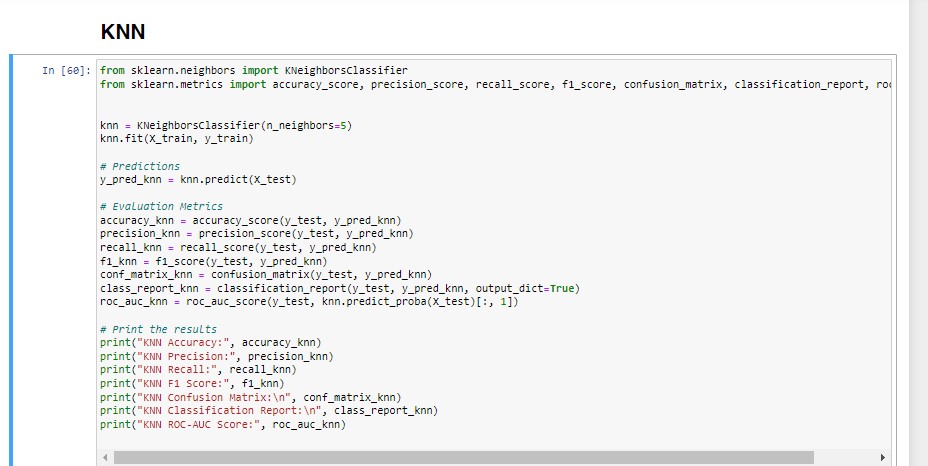
* + True Negatives (TN): 6889 (correctly predicted income ≤$50k)
  + False Positives (FP): 528 (incorrectly predicted income >$50k)
  + False Negatives (FN): 869 (incorrectly predicted income ≤$50k)
  + True Positives (TP): 1483 (correctly predicted income >$50k)



The Random Forest model performs reasonably well. There’s a particularly strong result for the majority class (class 0).

ii. KNN

A kNN classifier is defined with n\_neighbors=5. This indicates that the model will consider the 5 nearest neighbors for making the predictions.



## iia. KNN Results

-Accuracy: The accuracy here is about approx. 77.7%, This is the proportion of the total number of correct predictions.

-Precision for Class 1: Precision for the positive class (income > $50k) is about 56.7%, this would mean that when the model predicts an income > $50k, it is accurate 56.7% of the time.

-Recall for Class 1: Recall for the positive class is about 31.3%, this would mean that the model would correctly identify 31.3% of all actual cases of income > $50k.

-F1 Score for Class 1: F1 score of approximately 40.4% belonging to the positive class indicates a balance between precision and recall.

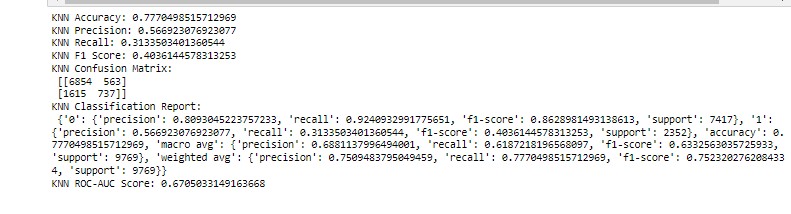
-Confusion Matrix: The confusion matrix here shows that the model has a higher tendency to predict correctly the negative class (income ≤ $50k) than the positive class.

-Classification Report: The results from the classification detailed metrics for each class. It indicates better performance in predicting the negative class when compared to the positive class.

-ROC-AUC Score: A score of approximately 67.1% suggests fair ability of the model to tell between classes.

Generally, this classifier indicates good performance especially in highlighting the majority class (i.e income less than $50k). Nonetheless, it appears to be a hassle to accurately classify the minority class (i.e income greater than $50k) as indicated by lower precision and recall for this class.

The ROC-AUC score also points to potential improvement areas.



### iii. Decision Tree

A Decision Tree classifier is defined with default parameters.

iiia.

Decision Tree

Results



-Accuracy (0.8044): The model will correctly predicts the outcome 80.44% of the time, This value indicates an overall good performance.

-Precision for Class 1 (0.5924): Approximately 59.24% of the time, when the model predicts an income greater than $50k (class 1), it is correct. This suggests moderate precision.

-Recall for Class 1 (0.6012): The model will correctly identify 60.12% of all instances of income greater than $50k. when the model's ability to find the majority of positive instances is taking into consideration, this be referred to as a fair recall rate.

-F1 Score for Class 1 (0.5968): The F1 score, which balances precision and recall, is about 59.68% for the positive class. This score is a measure of the model's accuracy in classifying the positive class (income > $50k).

-Confusion Matrix:

* 6444 instances of negative class (income ≤ $50k) were correctly identified (i.e. True

Negatives).

* 973 instances of negative class were incorrectly classified as positive (i.e. False Positives).
* 938 instances of positive class (income > $50k) were incorrectly classified as negative (i.e.

False Negatives).

* 1414 instances of the positive class were correctly identified (ie.True Positives).

-Classification Report:

* Class 0 (income ≤ $50k) has higher precision, recall, and F1 score when compared to class

1.

* Reflecting the overall performance across both classes when the weighted average scores take the class imbalance into account.

-ROC-AUC Score (0.7357):

* The ROC-AUC score is approximately 73.57%. This score is a measure of the model's ability to tell the difference between the two classes. A score closer to 1 shows better performance.

For this case, the score is moderate, suggesting that there’s some room for improvement.

Overall, this model demonstrates a good performance. The performance is noticed to be better in in identifying the negative class (income ≤ 50K), while lower performance in identifying positive class (income > 50K).

* **AML:**

To perform the required classification analysis, two classification algorithms will be applied and the performance evaluated.

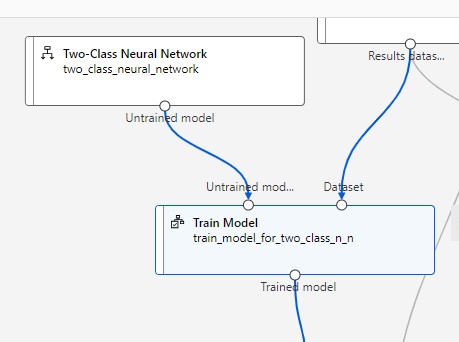
|  |  |
| --- | --- |
| **Algorithm** | **Selection Justification** |
| Two-Class Neural Network | Handles binary classification task, Ease of handling complex patterns in data, |
| Two-Class Support Vector  Machine (SVM) | Handles binary classification task, High-dimensional data handling, Diverse model evaluation |

i.

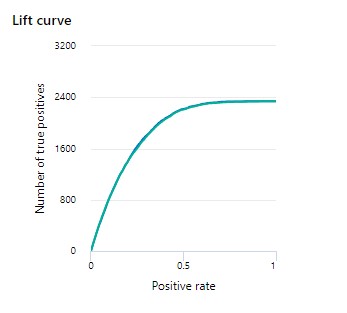
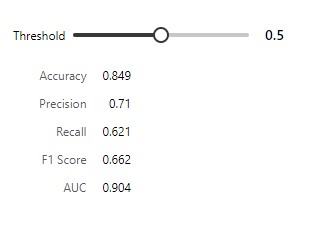
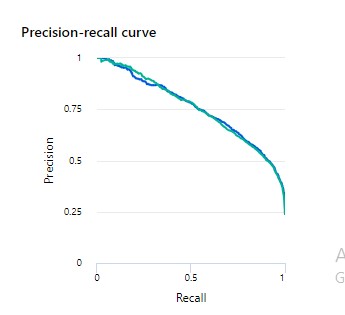
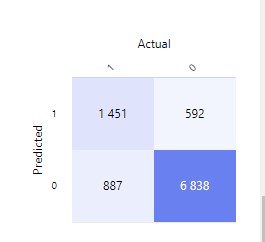
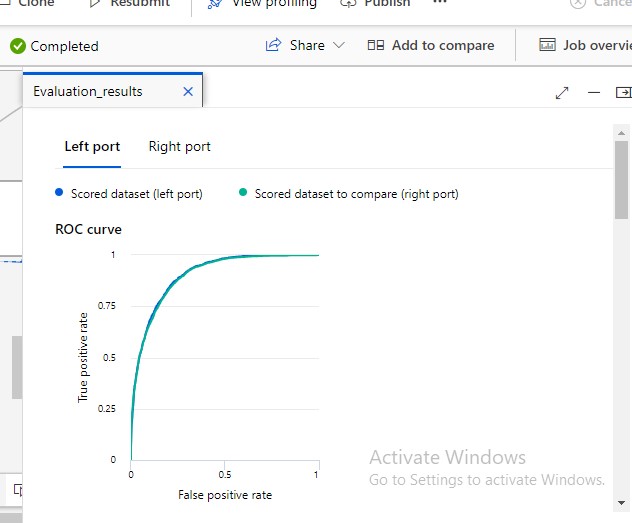
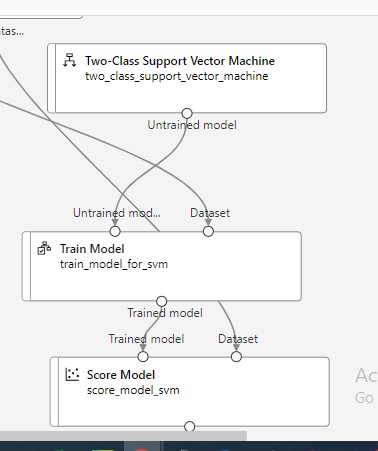
Two

-

Class Neural Network



## ia. Two-Class Neural Network Result



The Two-Class Neural Network's performance at a threshold of 0.5 indicates that:

-Accuracy (0.849): The model will correctly predict both classes about 84.9% of the time. Indicating overall good model performance .This is a high accuracy rate

-Precision (0.71): When the model predicts the positive class, it will be correct 71% of the time. This indicates a fairly high level of reliability in the model's positive predictions.

-Recall (0.621): The model will correctly identify 62.1% of all positive cases. This suggests a fair ability to detect positive instances.

-F1 Score (0.662): The achieved balance between precision and recall results in an F1 score of

66.2%, which is a measure of the model's accuracy in classifying the positive class.

-AUC (0.904): The Area Under the Curve (AUC) for the Receiver Operating Characteristic (ROC) is

90.4%, presenting a strong ability of the model to differentiatiate between the two classes.

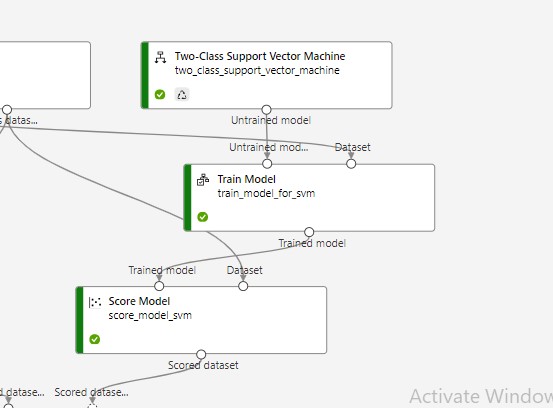
Overall, the Two-Class Neural Network shows a pretty high accuracy and a strong ability to distinguish between classes, with moderately good precision and recall, indicating a well-performing model for this binary classification task.

ii.

Two

-

Class Support Vector Machine (SVM)



## iia. Two-Class Neural Network Result

The Two-Class Support Vector Machine (SVM)'s performance at a threshold of 0.5 indicates that:

-Accuracy (0.847): The model will correctly predict both classes about 84.7% of the time. Indicating overall good model performance This is a high accuracy rate

-Precision (0.723): When the model predicts the positive class, it will be correct 72.3% of the time.

This indicates a fairly high level of reliability in the model's positive predictions.

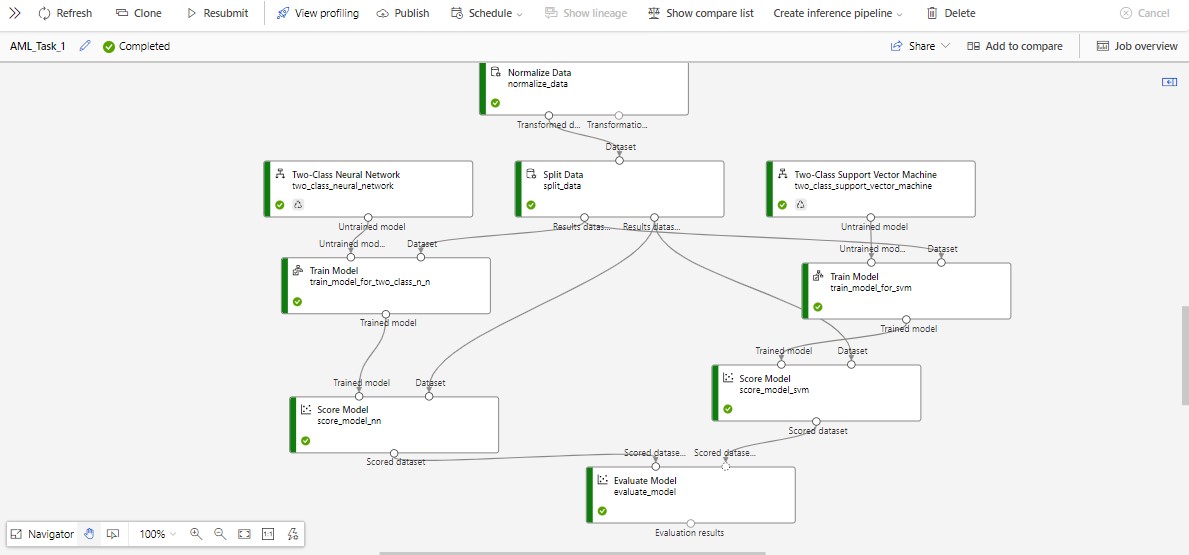
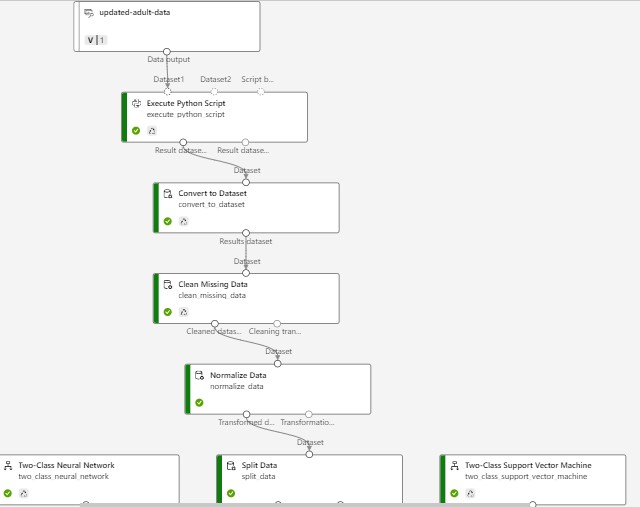
-Recall (0.589): The model will correctly identify 58.9% of all positive cases. This suggests a moderate ability to detect positive instances. Inferring that the model’s ability positive instance is a bit limited.

-F1 Score (0.649): The achieved balance between precision and recall results in an F1 score of 64.9%, which is a measure of the model's accuracy in classifying the positive class.

-AUC (0.9): The Area Under the Curve (AUC) for the Receiver Operating Characteristic (ROC) is

90.4%, presenting a strong ability of the model to differentiate between the two classes.

In summary, the Two-Class SVM demonstrates high accuracy and strong distinguishable ability with high AUC. It does show good precision but has room for improvement in recall, implying a need to better identify positive instances



# 6. Recommendation and Model Suitability

The ***Random Forest*** classifier is recommended given the task performed which involved predicting income levels using a relatively complex dataset with multiple features. Here's why:

-*High Accuracy and ROC-AUC Score*: Random Forest outperformed the other models in relations to overall accuracy and the ROC-AUC score. This shows its effectiveness in separating the two classes (i.e. income levels) and its maturity against overfitting. This is an important characteristic to have in a dataset with multiple variables.

-*Balance of Precision and Recall*: Even though its recall is slightly lower than the Decision Tree model, Random Forest still maintains a suitable balance between precision and recall. This is an important characteristic to have in ensuring that the model is not only good at identifying positive cases but also accurate in its predictions.

-*Handling Complex Relationships*: Random Forest is expert at handling complex relationships between features due to its aggregating nature. This is an important characteristic to have in a dataset which likely contains non-linear relationships and interactions between variables.

-*Generalization Capability*: The model's high ROC-AUC score suggests a strong generalization capability. It is likely to perform well not just on the test data but also on new, unseen data. This is an essential characteristic for real-world deployment.

-*Maturity to Overfitting*: Random Forest, by its very design (an aggregation of decision trees), is more robust to overfitting compared to a single Decision Tree, especially in cases where the dataset is large and varied.

Considerations Computational Resources: Random Forest may possibly need more computational resources because of its complexity. If this is a significant hold-back, the Decision Tree could be a secondary option. In addition, if the requirement for model interpretability outweighs the need for accuracy, the Decision Tree would be preferable due to its simplicity, less technical interpretable nature. For this task, however, there is validation to prioritize accuracy, robustness, and the ability to handle complex data structures makes Random Forest the most astute choice for future deployment.

On the consideration of Neural Network which shows promising results is recommended if the analysis would involve computation of complex and non-linear relationships and if computational resources and interpretability would not be primary constraints.

Potential Improvement: Some methods that may be expored for potential future improvements are listed below:

* Hyperparameter Tuning
* Feature Engineering
* Handling Class Imbalance
* Ensemble Methods