**Optimizing an ML Pipeline in Azure**

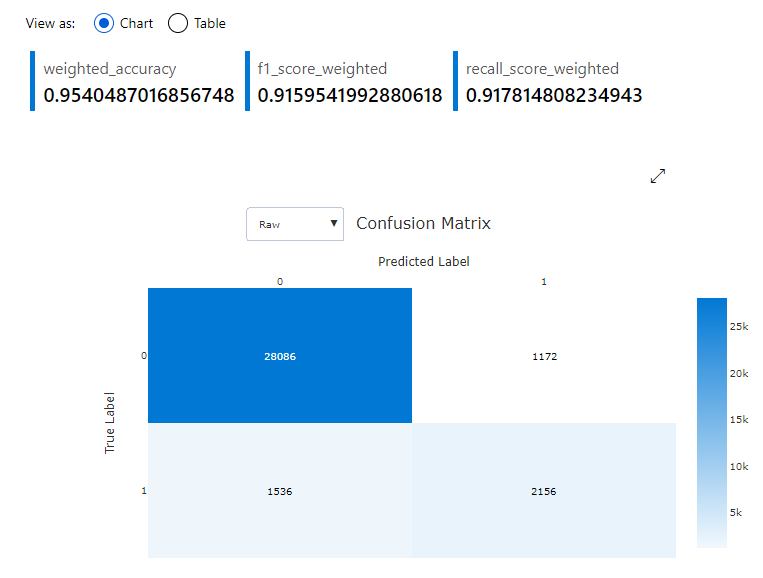
**Overview**

This project is part of the Udacity Azure ML Nanodegree. In this project, we build and optimize an Azure ML pipeline using the Python SDK and a provided Scikit-learn model. This model is then compared to an Azure AutoML run.

**Summary**

The dataset contains about 32950 rows × 21 columns which was collected by Portuguese marketing institution through marketing campaigns using phone calls. Using classification we wanted to predict weather a customer will subscribe or not based on this historic data.

The best performing model was a VotingEnsemble through AzureML with an accuracy of 91.78. The goal of **ensemble methods** is to combine the predictions of several base estimators built with a given learning algorithm in order to improve generalizability / robustness over a single estimator. The HyperDrive method has given an accuracy of 91.09 which is closed to the one using AzureML.



**Scikit-learn Pipeline**

Explain the pipeline architecture, including data, hyperparameter tuning, and classification algorithm.

Data was imported using TabularDatasetFactory. After feature engineering, 1 hot encoding was performed to convert non-numerical features to 1- hot encoding. The data was splitted to train and test datasets. Hyperdrive was used using SDK to perform hyperprameter tuning and then LogisticRegression was applied.

I have used Randon sampling. Random sampling supports discrete and continuous hyperparameters. It supports early termination of low-performance runs. Some users do an initial search with random sampling and then refine the search space to improve results.

For early stopping policy, I have used Bandit policy. Bandit policy is based on slack factor/slack amount and evaluation interval. Bandit terminates runs where the primary metric is not within the specified slack factor/slack amount compared to the best performing run.

**AutoML**

AutoML was configured as classification with primary metric for accuracy. The maximum timeout was defined as 30 minutes. The model after 45 child run have given accuracy for various algorithms and finally the VotingEnsemble has given as accuracy of 91.78 %

**Pipeline comparison**

AutoML has given an accuracy of 91.78% while HyperDrive has given a accuracy of 91.09%. Both the models has given almost similar accuracy as we can say that both are suitable for this problem.

In HyperDrive packages, Hyperparameters are adjustable parameters you choose for model training that guide the training process. The HyperDrive package helps you automate choosing these parameters. For example, you can define the parameter search space as discrete or continuous, and a sampling method over the search space as random, grid, or Bayesian. Also, you can specify a primary metric to optimize in the hyperparameter tuning experiment, and whether to minimize or maximize that metric. You can also define early termination policies in which poorly performing experiment runs are canceled and new ones started. If with minimal effort and closed to better results are the goal then we can straight forward use AutoML.

**Future work**

What are some areas of improvement for future experiments? Why might these improvements help the model?

Using different parameter sampling techniques i.e. Bayesian sampling and Grid sampling. The BanditPolicy parameters can also be adjusted for better performance.

As we see in the feature importance summary graph, we can see that that some of the features has more contribution to the negative classification. Some more positive classified data should be collected for training to get more balanced data set and it can also help in model performance.

