# Introduction

## Slide 1

Hello and welcome. This video is part of the second project for the Udacity Nanodegree Program, Machine Learning Engineering with Microsoft Azure.

The objective is to show how to operationalize machine learning

# Topics

## Slide 2

The topics that I will cover in this video are:

* An explanation of the business case that I am trying to solve through Machine Learning,
* The architecture used to solve the business case, and
* A demonstration of the steps followed:
  + The manual deployment of the model, and
  + The deployment through a pipeline to automate a part of the process

# Assumptions

## Slide 3

There are some topics that I will assume you are familiar with:

* Machine Learning concepts
* Azure Cloud Platform
* Azure Machine Learning Studio
* Azure AutoML
* Azure Command Line Interface
* Python
* Python SDK for Azure, and
* Jupyter notebooks

# Business case

## Slide 4

Let's start with the business case that I am trying to solve through Machine Learning

## Slide 5

The business case for this project uses the Bank Marketing dataset originally hosted by UCI

## URL

This dataset is related with direct marketing campaigns through phone calls, of a Portuguese banking institution.

The goal is to predict if the client will subscribe a term deposit

The dataset contains 20 client features, which are numerical or categorical

The desired target is the variable “y” that has a binary value of yes or no

## Slide 5

For the project, the training dataset was provided through the link shown.

There are also testing and validation datasets.

# Architecture

## Slide 6

Now, let’s see the architecture used to solve the business case

## Slide 7

For the manual deployment, the starting point is the dataset provided as a csv file

It is uploaded to Azure Machine Learning Studio as a tabular dataset

An AutoML experiment is created and executed to generate several classification models

The best model is selected, based on the accuracy obtained by each model

This model is deployed and Application Insights is enabled, to monitor the model

The deployed model is tested by interacting with the endpoints

For the pipeline deployment, a pipeline is created. A pipeline is an independent executable workflow of a machine learning task. The pipeline for this project is focused on the execution of an AutoML experiment

# Generate the Model

## Slide 8

Let’s start the demo with the generation of the model for the manual deployment

## Slide 9

This part of the demo will start with the dataset up to the generation of the models.

## Azure ML Studio

This is Azure ML Studio

## Dataset

To see the dataset uploaded, I go to the Dataset section and verify that it appears in my list.

To see its properties, I click on the dataset name and verify that it is a tabular dataset.

To get a preview of its contents, I click on Explore. I scroll to the right to make sure that the target column is present.

## Auto ML experiment

To see my completed experiments, I click on Experiments

Here I have the completed experiment, and I will click on it, to see the results.

I click on my first run.

The experiment shows a summary of the best model generated.

I will click on Models, to visualize all the models that were generated.

By default, AutoML experiments run Stack and Voting Ensembles as the last steps. So the experiments generated 46 experiments in one hour.

# Deploy the Model

## Slide 10

The next step is the deployment of the model

## Slide 11

This part of the demo continues with the deployment of the model and the enablement of Application Insights.

## Slide 12

Authentication is crucial for the continuous flow of operations. When authentication is not set properly, it requires human interaction and the flow is interrupted.

To ensure this continuous flow, I created a Service Principal, which is a user role with controlled permission to access specific resources.

The steps needed to create a Service Principal are:

* Login to the Azure cloud platform
* Add the Azure Machine Learning extension
* Create the Service Principal, and
* Assign access to the workspace and resource group

## Slide 13

This is the command line interface showing the login to the Azure cloud platform, with the az login command

## Slide 14

Here are the commands to install the Azure Machine Learning extension, create the Service Principal, and assign access to the workspace and resource group

Now, let’s switch to Azure ML Studio to deploy the model.

## Azure ML Studio

In the best model window, I click on Deploy

I provide a model name

bankmarketing-model-2

Provide a description

Bank Marketing deployed model

Select the compute type as Azure Container Instance, as requested for the project.

ACI uses container technology to quickly deploy compute instances and is simpler to use, compared to AKS

I will enable authentication, so the Service Principal created is used

In Advanced options, Application Insights can be enabled manually.

For the project, the request was to enable it through code after the deployment, so I will explain how it was performed after the deployment is done.

Once the proper settings are in place, the next step would be to click on Deploy.

I will cancel this deployment, since I already have a deployed model.

I click on the Model tab, to verify that my model is already deployed.

## Slide 15

This is the python script used to enable Application Insights and the command line interface used to execute it.

## Slide 16

This is the python script used to produce the logging output and the command line interface used to execute it.

## Azure ML Studio

Back in Azure ML Studio, I click on the deployed model name, which takes me to the endpoints section.

Here is the information related to the deployed model, including the enablement of Application Insights.

If I click the URL, it takes me to Application Insights, where I can monitor the model.

# Consume Endpoints

## Slide 17

The next step is the consumption of endpoints

## Slide 18

A deployed service can be consumed through an HTTP API. An HTTP API is a URL that is exposed over the network, so the interaction with a trained model can happened through HTTP requests.

Users can initiate an input request, usually via an HTTP POST request, which is used to submit data.

Users can request data via an HTTP GET request from a URL.

The API exposed by Azure Machine Learning uses JSON to accept data and submit responses.

The interaction with the endpoints can be performed via Swagger and Python scripts.

This part of the demo will show how to interact with the model through the endpoints.

## Slide 19

To interact with through Swagger, in the deployed model page Azure provides a Json file with the parameters of the model.

A script was provided to create an HTTP server to expose the json file.

## Slide 20

This is the Swagger HTTP server, with the GET and POST requests.

## Slide 21

To test the request, an input payload must be provided.

## Slide 22

And a response is provided. In this case, for the client features provided, the prediction was that the client will not respond to the term deposit.

Now, let’s see how to interact with the endpoints through Python scripts.

## Azure ML Studio

I will need the REST endpoint information of the model and the key to access it.

I click on the Endpoints section to get the deployed model.

Then I click on the deployed model.

I can get the REST endpoint in the Details tab of the model.

I can also find this information and the key in the Consume tab.

## Notepad++

Here I have the script that I will use to test the endpoints.

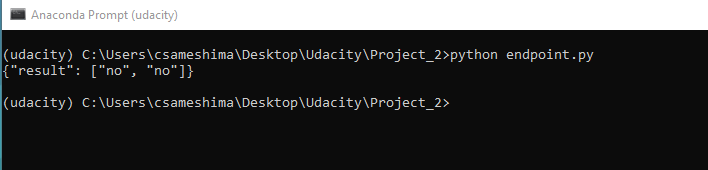
I have the scoring URI and the key obtained in the previous step.

The script has two sets of data to score with the characteristics of a client, converts them to JSON, sets the content type, uses the key provided, makes the POST request and displays the response, which is if the client will accept to the term deposit, or not.

Let’s switch to the command line interface to see it working

## CLI

I execute the script and get the response for both data sets. In this case both clients will not accept the term deposit.

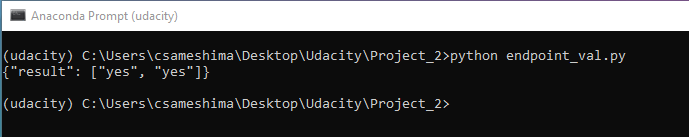


## Notepad++

I generated an additional script, with two sets of data from the validation dataset, which were labeled as yes.

## CLI

I execute the script and get the response for both data sets. In this case both clients will accept the term deposit.



This concludes the manual deployment of the model.

# Pipeline Deployment

## Slide 19

The second demonstration is the deployment of a pipeline, to automate part of the process

## Slide 20

Automation is a core pillar of DevOps, applicable to Machine Learning operations.

A workflow can be automated via pipelines.

For the project, a notebook was provided to generate the pipeline, to execute an AutoML experiment.

## Azure ML Studio

## Notebook

This is the notebook used to create a pipeline to automate the generation of an AutoML experiment.

The steps used are:

* Create an experiment in the workspace
* Attach a compute cluster
* Define the dataset
* Configure AutoML
* Set the execution of AutoML as a pipeline step and launch it
* Select the best model and train it
* Explore the results
* Test the best fitted model, and
* Publish the AutoML execution pipeline

First, the libraries are imported.

The Workspace is initialized.

Instead of creating a new experiment, the one already created is used.

The compute cluster already created is attached.

The dataset is loaded and examined.

The AutoML run is configured, created, and executed as a pipeline step.

Once finished, the metrics of the generated models are retrieved.

The best model is selected and saved.

The best model is tested with a different dataset and the confusion matrix is obtain to evaluate the results.

The last part is the publication of the pipeline. The name of the pipeline is Bankmarketing Train.

Now, I will show the published pipeline in the Pipeline section.

I select Pipeline endpoints and I see the published pipeline.

I click on the name and see the REST endpoint generated.

Back to the notebook, the published pipeline is triggered to schedule an AutoML experiment, named pipeline-rest-endpoint.

The pipeline is submitted. I click on the View run details, to monitor the run.

The run is completed.

To see the models generated I click on Steps. Then I click on the run number.

I get the status of the run, with the best model generated.

I click on the Models tab, to see the models that were generated by the experiment.

I click the best model and review its details.

I already deployed the model, so I click on the Model tab, to get the deployed model.

To test that the model is working, I get the REST endpoint and key values in the Consume tab.

## Notepad++

Here I have the script that I will use to test the endpoints of the pipeline generated model.

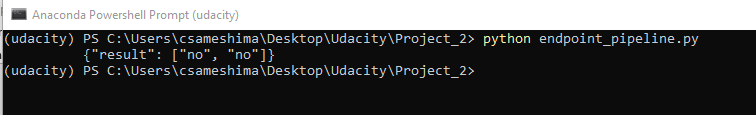
It is the same as the one used before to test the endpoints, but I am using the scoring URI and key for the pipeline generated model.

I also generated a second script, with the sets of data from the validation dataset, which were labeled as yes.

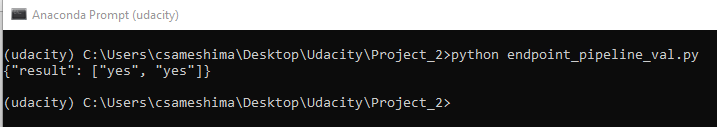
Let’s switch to the command line interface to see them working

## CLI

I execute the first script and get the response for both data sets, which is negative.



I execute the second script and get the response for both datasets, which is positive.



# End

## Slide 21

With this, I finish the presentation of my project.

Thank you very much for your attention.

Stay safe and healthy.