Azure AI Fundamentals Notes

Notes and Procedures

Bruttocao, Marco (858067)*

Contents

*858067@stud.unive.it

Get started with artificial intelligence on Azure

Azure Machine Learning service is a **cloud-based** platform for creating, managing, and publishing machine learning models:

- **Automated machine learning**: To quickly create an effective machine learning model from data
- **Azure Machine Learning designer**: A graphical interface enabling no-code development
- **Data and compute management**: Cloud-based data storage and compute resources
- **Pipelines**: define pipelines to orchestrate model training, deployment, and management tasks

Anomaly detection

In Microsoft Azure, the **Anomaly Detector** service provides an application programming interface (API) that developers can use to create anomaly detection solutions.

Computer Vision models and capabilities

- **Image classification**: Machine learning models trained to classify images based on their contents
- **Object detection**: Machine learning models trained to classify individual objects within an image, and identify their location with a bounding box
- **Semantic segmentation**: It is an advanced machine learning technique in which individual pixels in the image are classified according to the object to which they belong
- **Image analysis**: It extracts information from images, including "tags" or even descriptive captions
- **Face detection**: It is a specialized form of object detection that locates human faces in an image
- **Optical character recognition**: It is a technique used to detect and read text in images

Understand natural language processing

- Analyze and interpret text
- Interpret spoken language, and synthesize speech responses
- Automatically translate spoken or written phrases between languages
- Interpret commands and determine appropriate actions

Understand conversational AI

Bots can be the basis of AI solutions for:

- Customer support
- Reservation systems
- Health care consultations
- Home automation and personal digital assistants

At Microsoft, AI software development is guided by a set of *six principles*, designed to ensure that AI applications provide amazing solutions to difficult problems *without any unintended negative consequences*:

- **Fairness**
- **Reliability and safety**
- **Privacy and security**
- **Inclusiveness**
- **Transparency**
- **Accountability**

Use visual tools to create machine learning models with Azure Machine Learning

Azure Machine Learning

Azure Machine Learning is a cloud service that you can use to train and manage machine learning models. It includes a wide range of features and capabilities that help data scientists prepare data, train models, publish predictive services, and monitor their usage. Most importantly, it helps data scientists increase their efficiency by automating many of the time-consuming tasks associated with training models; and it enables them to use *cloud-based compute resources* that scale effectively to handle large volumes of data while incurring costs only when actually used.

Create a workspace:

- Sign into the Azure portal
- Create a resource, search for Machine Learning
- Subscription: Your Azure subscription
- Resource group: Create or select a resource group
- Workspace name: Enter a unique name for your workspace
- Region: Select the geographical region closest to you

There are four kinds of compute resource you can create (In Azure Machine Learning studio, view the Compute page -under Manage-):

- **Compute Instances**: Development workstations to work with data and models
- **Compute Clusters**: Scalable clusters of virtual machines for on-demand processing of experiment code
- **Inference Clusters**: Deployment targets for predictive services that use trained models
- **Attached Compute**: Links to existing Azure compute resources, such as Virtual Machines or Azure Databricks clusters.

Create compute targets:
On the Compute Instances tab, add a new compute instance

Create compute clusters:
Compute Clusters tab, add a new compute cluster

Create a dataset

In Azure Machine Learning, data for model training and other operations is usually encapsulated in an object called a **dataset**.

Datasets:
Create a new dataset from local files

Run an automated machine learning experiment

In Azure Machine Learning, operations that you run are called **experiments**. Follow the steps below to run an experiment that uses automated machine learning to train a regression model.

Automated ML page:
Create a new Automated ML run

Deploy a model as a service

You can deploy the best performing model as a service for client applications to use: you can deploy a service as an Azure Container Instances (ACI) or to an Azure Kubernetes Service (AKS) cluster. For production scenarios, an AKS deployment is recommended, for which you must create an inference cluster compute target.

Clean-up

The web service you created is hosted in an Azure Container Instance. If you don't intend to experiment with it further, you should *delete the endpoint* to avoid accruing unnecessary Azure usage. You should also *stop the training cluster and compute instance resources* until you need them again.

- In Azure Machine Learning studio, on the **Endpoints** tab, then select **Delete** and confirm that you want to delete the endpoint
- On the **Compute page**, on the **Compute Instances** tab, select your compute instance and then select **Stop**

Create a Regression Model with Azure Machine Learning designer

You can use Microsoft Azure Machine Learning designer to create regression models by using a *drag and drop visual interface*, without needing to write any code. To use the Azure Machine Learning designer, you create a **pipeline** that you will use to train a machine learning model. This pipeline starts with the dataset from which you want to train the model.

Add an Evaluate Model module

In the **Model Scoring & Evaluation** section, drag an **Evaluate Model** module to the canvas, under the Score Model module. These include the following regression performance metrics:

- **Mean Absolute Error** (MAE): The average difference between predicted values and true values
- **Root Mean Squared Error** (RMSE): The square root of the mean squared difference between predicted and true values
- **Relative Squared Error** (RSE): A relative metric between 0 and 1 based on the square of the differences between predicted and true values
- **Relative Absolute Error** (RAE): A relative metric between 0 and 1 based on the absolute differences between predicted and true values
- **Coefficient of Determination** (R2): This metric summarizes how much of the variance between predicted and true values is explained by the model

Create and run an inference pipeline

- In Azure Machine Learning Studio, click the Designer page to view all of the pipelines you have created
- In the Create inference pipeline drop-down list, click Real-time inference pipeline

Evaluate a classification model

The **confusion matrix** shows cases where both the predicted and actual values were 1 (known as *true positives*) at the top left, and cases where both the predicted and the actual values were 0 (*true negatives*) at the bottom right. The other cells show cases where the predicted and actual values differ (*false positives* and *false negatives*).

- **Accuracy**: The ratio of *correct predictions* (true positives + true negatives) to the *total number of predictions*
- **Precision**: The fraction of *positive cases correctly identified* (the number of true positives divided by the number of true positives plus false positives)
- **Recall**: The fraction of the *cases classified as positive that are actually positive* (the number of true positives divided by the number of true positives plus false negatives)
- **F1 Score**: An overall metric that essentially combines precision and recall.
- **ROC curve** (received operator characteristic): Another term for recall is True positive rate, and it has a corresponding metric named False positive rate, which measures the number of negative cases incorrectly identified as positive compared the number of actual negative cases. Plotting these metrics against each other for every possible threshold value between 0 and 1 results in a curve. In an ideal model, the curve would go all the way **up the left side and across the top**, so that it covers the full area of the chart. The larger the area under the curve (which can be any value from 0 to 1), the better the model is performing

Evaluate a clustering model

- **Average Distance to Other Center**: This indicates how close, on average, each point in the cluster is to the centroids of all other clusters
- **Average Distance to Cluster Center**: This indicates how close, on average, each point in the cluster is to the centroid of the cluster
- **Number of Points**: The number of points assigned to the cluster
- **Maximal Distance to Cluster Center**: The maximum of the distances between each point and the centroid of that point's cluster. If this number is high, the cluster may be widely dispersed.

Explore computer vision in Microsoft Azure

In Microsoft Azure, the Computer Vision cognitive service uses pre-trained models to analyze images, enabling software developers to easily build applications that can:

- Interpret an image and suggest an appropriate caption
- Suggest relevant tags that could be used to index an image
- Categorize an image
- Identify objects in an image
- Detect faces and people in an image
- Recognize celebrities and landmarks in an image
- Read text in an image

For each of these tasks, you need a resource in your Azure subscription. You can use the following types of resource:

- **Custom Vision**: A dedicated resource for the custom vision service, which can be either a training, a prediction or a both resource
- **Cognitive Services**: A general cognitive services resource that includes Custom Vision along with many other cognitive services. You can use this type of resource for training, prediction, or both

Whichever type of resource you choose to create, it will provide two pieces of information that you will need to use it:

- A *key* that is used to authenticate client applications
- An *endpoint* that provides the **HTTP address** at which your resource can be accessed.

Features:

- **Tagging visual features**: the image descriptions generated by Computer Vision are based on a set of thousands of recognizable objects, which can be used to suggest tags for the image
- **Detecting objects**: not only will you get the type of object, but you will also receive a set of coordinates that indicate the top, left, width, and height of the object detected
- **Detecting brands**: if a known brand is detected, the service returns a response that contains the brand name, a confidence score (from 0 to 1 indicating how positive the identification is), and a bounding box (coordinates) for where in the image the detected brand was found
- **Detecting faces**: the Computer Vision service can detect and analyze human faces in an image, including the ability to determine age and a bounding box rectangle for the location of the face(s).
- **Categorizing an image**: computer Vision can categorize images based on their contents. The service uses a parent/child hierarchy with a "current" limited set of categories
- **Detecting domain-specific content**: When categorizing an image, the Computer Vision service supports two specialized domain models:
  - Celebrities: The service includes a model that has been trained to identify thousands of well-known celebrities from the worlds of sports, entertainment, and business
  - Landmarks: The service can identify famous landmarks
- **Optical character recognition**: the Computer Vision service can use optical character recognition (OCR) capabilities to detect printed and handwritten text in images.

Azure resources for Custom Vision (also valid for object detection)

Creating an image classification solution with Custom Vision consists of two main tasks. First you must use existing images to *train the model*, and then you must *publish the model* so that client applications can use it to generate predictions.

For each of these tasks, you need a resource in your Azure subscription. You can use the following types of resource to access Custom Vision services:

- **Custom Vision**
- **Cognitive Services**

> When to use a Cognitive Services resource
> The simplest approach is to use a general Cognitive Services resource for both training and prediction. This means you only need to concern yourself with one *endpoint* (the HTTP address at which your service is hosted) and *key* (a secret value used by client applications to authenticate themselves).
>
> When to use a Custom Vision resource
> When you use a Custom Vision resource, you can track usage *specifically connected* to the Custom Vision service. However, you won't be able to use its endpoint and key interchangeably with applications leveraging other Cognitive Services.

To use your model, client application developers need the following information:

- **Project ID**: The *unique ID* of the Custom Vision project you created to train the model
- **Model name**: The name you assigned to the model during publishing
- **Prediction endpoint**: The *HTTP address* of the endpoints for the prediction resource to which you published the model (not the training resource)
- **Prediction key**: The *authentication key* for the prediction resource to which you published the model (not the training resource)

Detect and analyze faces with the Face service

Microsoft Azure provides multiple cognitive services that you can use to detect and analyze faces, including:

- **Computer Vision**: Which offers face detection and some *basic face analysis*, such as determining age
- **Video Indexer**: Which you can use to detect and identify faces in a video
- **Face**: which offers pre-built algorithms that can detect, recognize, and analyze faces

> Face currently supports the following functionality:
>
> - Face Detection
> - Face Verification
> - Find Similar Faces
> - Group faces based on similarities
> - Identify people

Face can return the rectangle coordinates for any human faces that are found in an image, as well as a series of attributes related to those faces such as age, blur, emotion, exposure, facial hair, glasses, hair, head pose, makeup, noise, occlusion and smile.

To use Face, you must create one of the following types of resource in your Azure subscription:

- **Face**: Use this specific resource type if you *don't intend* to use any other cognitive services, or if you want to *track utilization and costs for Face separately*
- Cognitive Services: Use this resource type if you plan to use multiple cognitive services and want to simplify administration and development.

Whichever type of resource you choose to create, it will provide two pieces of information that you will need to use it:

- A *key* that is used to authenticate client applications.
- An *endpoint* that provides the **HTTP address** at which your resource can be accessed.

  Tips for more accurate results

  - **Image format**: Supported images are JPEG, PNG, GIF, and BMP
  - **File size**: 6 MB or smaller
  - **Face size range**: From 36 x 36 up to 4096 x 4096. Smaller or larger faces will not be detected
  - **Other issues**: Face detection can be impaired by extreme face angles, occlusion (objects blocking the face such as sunglasses or a hand). Best results are obtained when the faces are full-frontal or as near as possible to full-frontal

Read text with the Computer Vision service

The **OCR API** is designed for quick extraction of *small amounts of text* in images. It operates synchronously to provide immediate results, and can recognize text in numerous languages.

When you use the OCR API to process an image, it *returns a hierarchy of information* that consists of:

- Regions in the image that contain text
- Lines of text in each region
- Words in each line of text

For each of these elements, the OCR API also returns *bounding box coordinates* that define a rectangle to indicate the location in the image where the region, line, or word appears.

The **Read API** is a better option for scanned documents that have a *lot of text*. The Read API also has the ability to automatically determine the proper recognition model to use, taking into consideration lines of text and supporting images with printed text as well as recognizing handwriting.

Because the Read API can work with larger documents, it works **asynchronously** so as not to block your application while it is reading the content and returning results to your application. This means that to use the Read API, your application must use a three-step process:

- Submit an image to the API, and retrieve an *operation ID* in response
- Use the operation ID to check on the *status of the image analysis* operation, and wait until it has completed
- *Retrieve the results* of the operation

The results from the Read API are arranged into the following hierarchy:

- Pages: One for each page of text, including information about the page size and orientation.
- Lines: The lines of text on a page.
- Words: The words in a line of text.

Each line and word includes bounding box coordinates indicating its position on the page.

Explore natural language processing

Analyze text with the Text Analytics service

There are some commonly used techniques that can be used to build software to analyze text, including:

- *Statistical analysis* of terms used in the text
- Extending *frequency analysis* to multi-term phrases, commonly known as N-grams
- Applying *stemming* or *lemmatization* algorithms to normalize words before counting them
- Applying linguistic structure rules to *analyze sentences*
- *Encoding* words or terms as numeric features that can be used to train a machine learning model
- Creating vectorized models that capture *semantic relationships* between words by assigning them to locations in n-dimensional space

You can choose to provision either of the following types of resource:

- A Text Analytics resource
- A Cognitive Services resource

Language detection

Use the language detection capability of the Text Analytics service to *identify the language* in which text is written. You can submit *multiple documents* at a time for analysis. For each document submitted to it, the service will detect:

- The **language name** (for example "English")
- The **ISO 6391 language code** (for example, "en")
- A **score indicating a level of confidence** in the language detection

Sentiment analysis

The Text Analytics service can evaluate text and return *sentiment scores and labels* for each sentence.

Using the pre-built machine learning classification model, the service evaluates the text and returns a sentiment score in the range of 0 to 1, *with values closer to 1 being a positive sentiment*. Scores that are close to the middle of the range (0.5) are considered *neutral* or *indeterminate*.

Key phrase extraction

Key phrase extraction is the concept of evaluating the text of a document, or documents, and then identifying the *main talking points* of the document(s).

Entity recognition

You can provide the Text Analytics service with *unstructured text* and it will return a *list of entities* in the text that it recognizes. The service can also provide links to more information about that entity on the web. An entity is essentially an item of a particular type or a category.

Speech recognition

Speech recognition is concerned with taking the spoken word and converting it into data that can be processed - often by transcribing it into a text representation. The spoken words can be in the form of a recorded voice in an audio file, or live audio from a microphone. Microsoft Azure offers both speech recognition and speech synthesis capabilities through the Speech cognitive service, which includes the following application programming interfaces (APIs):

- **The Speech-to-Text API**
- **The Text-to-Speech API**

You can use the speech-to-text API to perform real-time or batch transcription of audio into a text format. The audio source for transcription can be a real-time audio stream from a microphone or an audio file.

Microsoft Azure provides cognitive services that support translation. Specifically, you can use the following services:

- The **Translator Text service**, which supports text-to-text translation
- The **Speech service**, which enables speech-to-text and speech-to-speech translation

Translate text and speech

The Translator Text service is easy to integrate in your applications, websites, tools, and solutions. The service uses a *Neural Machine Translation* (NMT) model for translation, which analyzes the semantic context of the text and renders a more accurate and complete translation as a result.

> Translator Text service language support
> The Text Translator service supports text-to-text translation between more than 60 languages. When using the service, you_ must specify the language you are translating from and the language you are translating to using ISO 639-1 language codes_, such as en for English, fr for French, and zh for Chinese. Alternatively, you can specify cultural variants of languages by extending the language code with the appropriate 3166-1 cultural code - for example, en-US for US English, en-GB for British English, or fr-CA for Canadian French.

When using the Text Translator service, you can specify one from language with *multiple to languages*, enabling you to simultaneously translate a source document into multiple languages. The Translator Text API offers some optional configuration to help you fine-tune the results that are returned, including:

- **Profanity filtering**: Without any configuration, the service will translate the input text, without filtering out profanity. Profanity levels are typically culture-specific but you can control profanity translation by either marking the translated text as profane or by omitting it in the results.
- **Selective translation**: You can tag content so that it isn't translated. For example, you may want to tag code, a brand name, or a word/phrase that doesn't make sense when localized.

Speech translation with the Speech service

The Speech service includes the following application programming interfaces (APIs):

- **Speech-to-text**: used to transcribe speech from an audio source to text format.
- **Text-to-speech**: used to generate spoken audio from a text source.
- **Speech Translation**: used to translate speech in one language to text or speech in another.

Speech service language support
As with the Translator Text service, you can specify one source language and *one or more target languages* to which the source should be translated. You can translate speech into over 60 languages. The *source language must be specified* using the extended language and culture code format, such as es-US for American Spanish. This requirement helps ensure that the source is understood properly, allowing for localized pronunciation and linguistic idioms. The target languages must be specified using a two-character language code, such as *en* for English or *de* for German.

Create a language model with Language Understanding

Creating a language understanding application with Language Understanding consists of two main tasks. First you must define **entities**, **intents**, and **utterances** with which to *train the language model* - referred to as **authoring the model**. Then you must **publish the model** so that client applications can use it for intent and entity prediction based on user input. For each of the authoring and prediction tasks, you need a *resource* in your Azure subscription. You can use the following types of resource:

- **Language Understanding**: A dedicated resource for Language Understanding, which can be either an authoring or a prediction resource
- **Cognitive Services**: A general cognitive services resource that includes Language Understanding along with many other cognitive services. You can only use this type of resource for prediction

The separation of authoring and prediction resources is useful when you want to *track resource utilization for language model training separately from client applications* using the model to generate predictions. If you choose to create a Language Understanding resource, you will be prompted to choose authoring, prediction, or both - and it's important to note that if you choose "both", then two resources are created - one for authoring and one for prediction. Alternatively, you can use a dedicated Language Understanding resource for authoring, but deploy your model to a generic Cognitive Services resource for prediction. When your client application uses other cognitive services in addition to Language Understanding, this approach enables you to manage access to all of the cognitive services being used, including the Language Understanding prediction service, through a single endpoint and key.

Authoring

After you've created an *authoring resource*, you can use it to author and train a Language Understanding application by defining the **entities** and **intents** that your application will predict as well as **utterances** for each intent that can be used to train the predictive model. Language Understanding provides a comprehensive collection of prebuilt domains that include pre-defined intents and entities for common scenarios; which you can use as a starting point for your model. You can also create your own entities and intents.

Creating intents

Define intents based on actions a user would want to perform with your application. For each intent, you should include a variety of **utterances** that provide examples of how a user might express the intent. If an intent can be applied to multiple entities, be sure to include sample utterances for each potential entity; and ensure that each entity is identified in the utterance.

There are four types of entities:

- **Machine-Learned**: Entities that are *learned by your model* during training from context in the sample utterances you provide
- **List**: Entities that are *defined as a hierarchy* of lists and sublists
- **RegEx**: Entities that are *defined as a regular expression* that describes a pattern
- **Pattern.any**: Entities that are *used with patterns to define complex entities* that may be hard to extract from sample utterances

Get started with QnA Maker and Azure Bot Service

You can easily create a user support bot solution on Microsoft Azure using a combination of two core technologies:

- **QnA Maker**: This cognitive service enables you to create and publish a knowledge base with built-in natural language processing capabilities
- **Azure Bot Service**: This service provides a framework for developing, publishing, and managing bots on Azure

Conversations typically take the form of messages exchanged in turns. This pattern forms the basis for many user support bots, and can often be based on existing *FAQ documentation.* To implement this kind of solution, you need:

- A **knowledge** base of question and answer pairs
- A **bot service** that provides an *interface to the knowledge base* through one or more channels

Creating a QnA Maker knowledge base

The service provides a dedicated QnA Maker portal web-based interface that you can use to create, train, publish, and manage knowledge bases (you must first provision a QnA Maker resource in your Azure subscription). You can use the QnA Maker portal to *create a knowledge base* that consists of question-and-answer pairs. These questions and answers can be:

- Generated from an existing FAQ document or web page
- Imported from a pre-defined chit-chat data source
- Entered and edited manually

After creating a set of question-and-answer pairs, you must *train* your knowledge base. This process analyzes your literal questions and answers and applies a built-in natural language processing model to *match appropriate answers to questions*, even when they are not phrased exactly as specified in your question definitions. When you're satisfied with your trained knowledge base, you can publish it so that client applications can use it over its *REST interface.* To access the knowledge base, client applications require:

- The knowledge base **ID**
- The knowledge base **endpoint**
- The knowledge base **authorization key**

Build a bot with the Azure Bot Service

You can create a custom bot by using the *Microsoft Bot Framework SDK to write code* that controls conversation flow and integrates with your QnA Maker knowledge base. However, an easier approach is to use the *automatic bot creation* functionality of QnA Maker, which enables you create a bot for your published knowledge base and publish it as an Azure Bot Service application with just a few clicks. After creating your bot, you can manage it in the Azure portal, where you can:

- **Extend the bot's functionality** by adding custom code.
- **Test the bot** in an interactive test interface.
- **Configure logging, analytics, and integration** with other services.

When your bot is ready to be delivered to users, you can connect it to multiple channels; making it possible for users to interact with it through web chat, email, Microsoft Teams, and other common communication media.