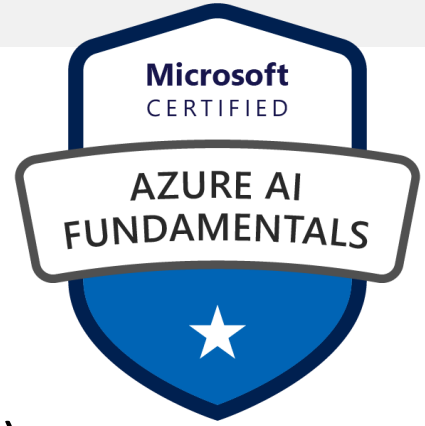
Azure AI Fundamentals (AI-900)

Lecture Slides

Good luck on your exam!

What is the Microsoft Azure AI Fundamentals (AI-900)?

Cheat sheets, Practice Exams and Flash cards 👉 www.exampro.co/ai-900



The Azure AI Fundamentals Certification is for those seeking an ML role such as **AI Engineer, or Data Scientist**

The certification will demonstrate a person can define and understand:

- Azure AI Services (e.g., Cognitive Services and Azure Applied AI Services)
- AI Concepts, Knowledge Mining, Responsible AI
- Basics of ML pipelines, Classical ML models, AutoML
- Generative AI workloads and Azure AI Studio

This certification is generally referred to by its course code **AI-900**

The **natural path** for the **Azure AI Engineer** or the **Azure Data Scientist**

This is an **easy course to pass**, and great for those **new to cloud or ML-related technology**

The Azure Roadmap

Cheat sheets, Practice Exams and Flash cards 👉 www.exampro.co/ai-900

Fundamentals

AZ-900

AI-900

DP-900

SC-900

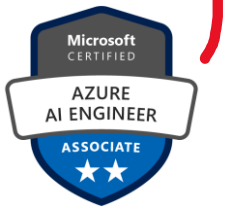
Associate

Administrator
- AZ-104

Developer
- AZ-204

Security
- AZ-500

Data Scientist
DP-100



AI Engineer
- AI-102

Data Engineer
- DP-203

Data Analyst
- DA-100

Expert

Solutions Architect Expert
- AZ-305

DevOps Engineer Expert
- AZ-400

Specialty

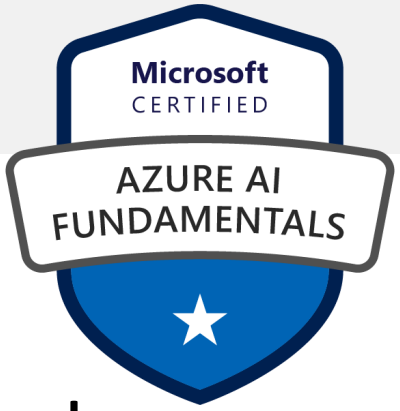
Azure for SAP Workloads
- AZ-120

IoT Developer
- AZ-220

Virtual Windows Desktop
- AZ-140

How Long to Study to Pass?

Cheat sheets, Practice Exams and Flash cards 👉 www.exampro.co/ai-900



Beginner
You are **New to ML and AI**
You are **New to Azure** or any cloud provider

Intermediate
You have passed
AZ-900 or DP-900
You are familiar with Azure

Experienced
You have **1+ Year Azure experience or another CSP e.g., AWS, GCP**

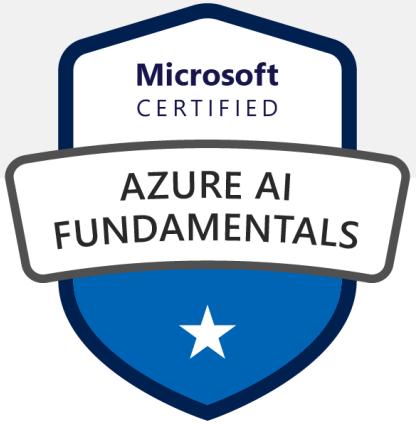
8 hours (average study time)

- 50% lecture and labs
- 50% practice exams

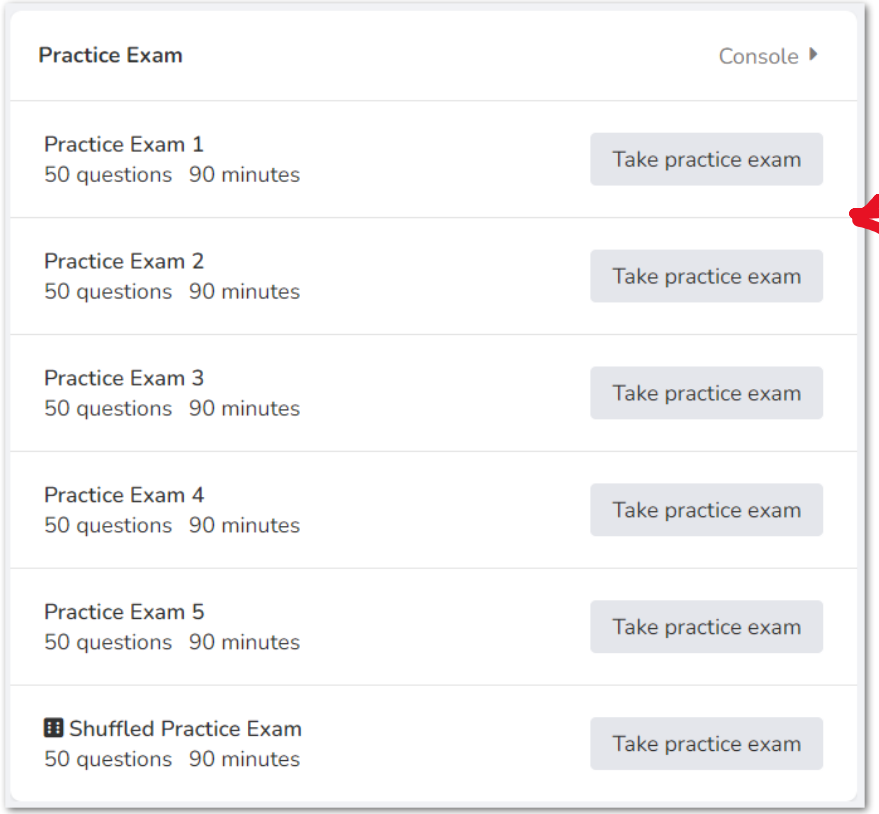
Recommended to study **30 min to 1 hour** a day for 14 days.

What does it take to pass the exam?

Cheat sheets, Practice Exams and Flash cards 👉 www.exampro.co/ai-900



1. **Watch video lecture** and memorize key information
2. **Do hands-on labs** and **follow along** within your own account
3. **Do paid online practice exams** that simulate the real exam.



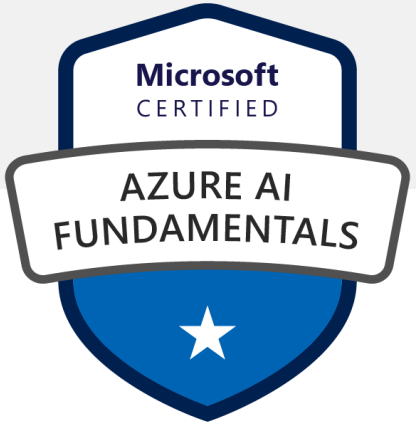
Exam Pro

Signup and Redeem your **FREE** Practice Exam
No credit card required

https://www.exampro.co/ai-900

Exam Guide – Content Outline

Cheat sheets, Practice Exams and Flash cards 👉 www.exampro.co/ai-900



The exam of questions in **5 domains**.

Each domain has its own weighting, this determines how many questions in a domain that will show up.

Skills measured

15-20% Describe Artificial Intelligence workloads and considerations

- 6-9 questions

20-25% Describe fundamental principles of machine learning on Azure

- 7-12 questions

15-20% Describe features of computer vision workloads on Azure

- 6-9 questions

15-20% Describe features of Natural Language Processing (NLP) workloads on Azure

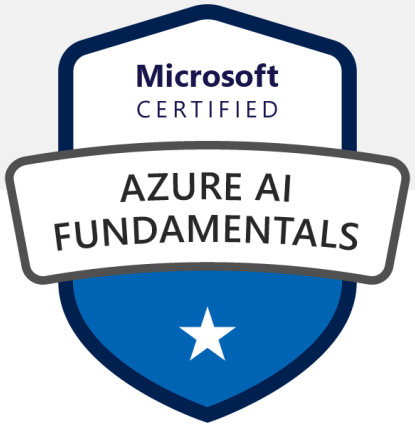
- 6-9 questions

15-20% Describe features of generative AI workloads on Azure

- 6-9 questions

Where do you take the exam?

Cheat sheets, Practice Exams and Flash cards 👉 www.exampro.co/ai-900



CERTIPORT
A PEARSON VUE BUSINESS
AUTHORIZED TESTING CENTER

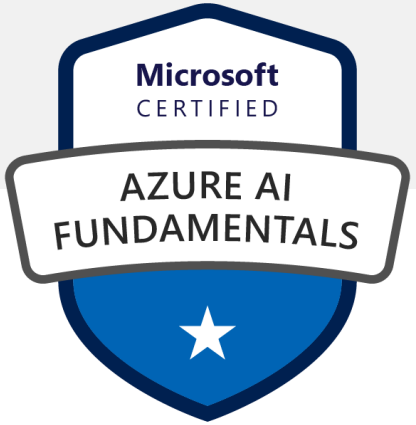
Pearson VUE

At an **in-person test center** or **online from the convenience of your own home**.

"proctor" is a supervisor, or person who monitors students during an examination.

Exam Guide – Grading

Cheat sheets, Practice Exams and Flash cards 👉 www.exampro.co/ai-900



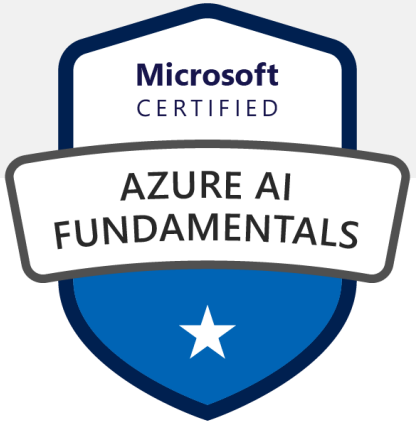
Passing Grade is *700/1000

You need to get "*around*" 70% to pass

Azure uses Scaled Scoring

Exam Guide – Response Types

Cheat sheets, Practice Exams and Flash cards 👉 www.exampro.co/ai-900



There are ~**37-47 Questions**

You can afford to get *10-13 questions wrong

There is **no penalty** for wrong questions

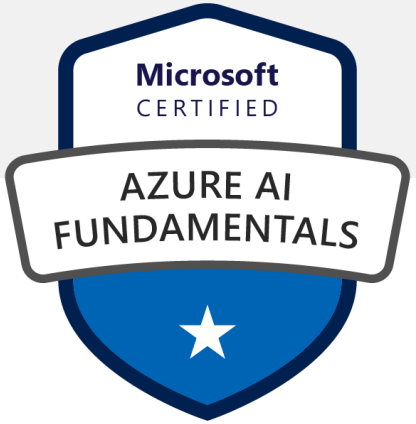
- Some question are worth **more than 1 point**
- Some questions cannot be skipped

Format of Questions

- Multiple Choice
- Multiple Answer
- Drag and Drop
- Hot Area
- Case Studies

Exam Guide – Duration

Cheat sheets, Practice Exams and Flash cards 👉 www.exampro.co/ai-900



Duration of **60 mins**

You get ~1mins per question

Exam Time is: **60mins**

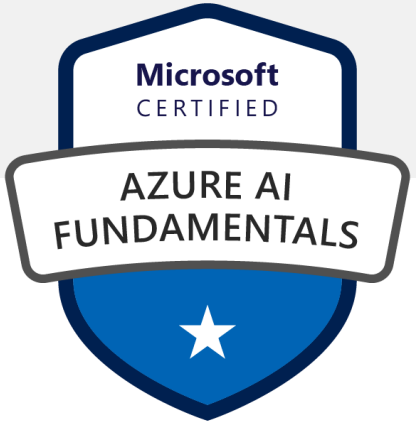
Seat Time is: **90mins**

Seat time refer to the amount of that you should allocate for the exam.
It includes:

- Time to review instructions
- Read and accept NDA
- Complete the exam
- Provide feedback at the end.

Exam Guide – Valid Until

Cheat sheets, Practice Exams and Flash cards 👉 www.exampro.co/ai-900



Valid forever

This certification does not expire

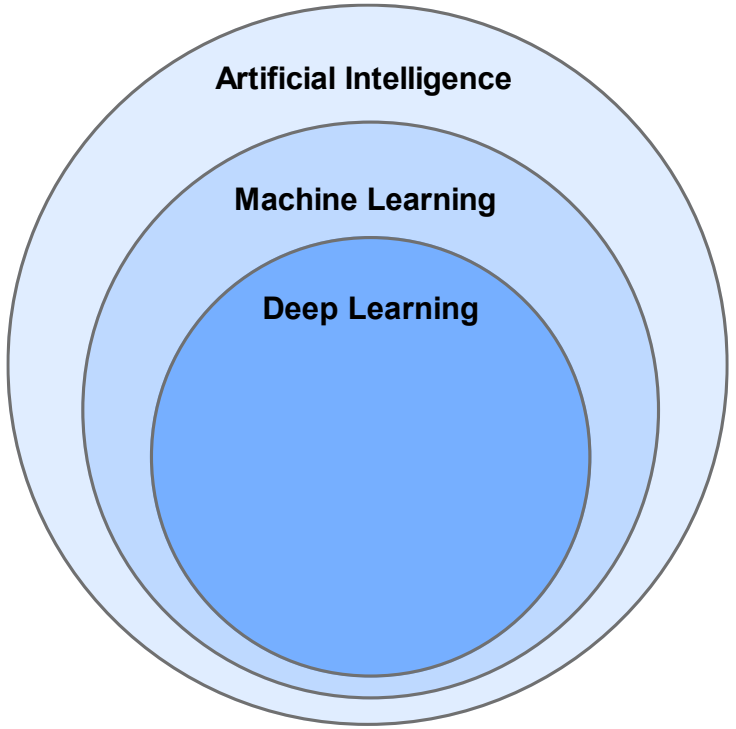
Microsoft Fundamentals certifications such as the AZ-900 or MS-900 **do not expire** as long as the technology is still available or relevant.

The Layers of Machine Learning

What is Artificial Intelligence (AI)?
Machines that perform jobs that **mimic human behavior**

What is Machine Learning (ML)?
Machines that get better at a task **without explicit programming**

What is Deep Learning (DL)?
Machines that have an **artificial neural network** inspired by the human brain to solve complex problems.



What is a Data Scientist?
A person with **multi-disciplinary skills** in math, statistics, predictive modeling and machine learning **to make future predictions**

Key Elements of AI

AI is the software that **imitates human behaviors and capabilities**.

Key elements *(according to Microsoft/Azure):*

- Machine learning — the foundation of an AI system, learn and predict like a human
- Anomaly detection — detect outliers or things out of place like a human
- Computer vision — be able to see like a human
- Natural language processing — be able to process human languages and infer context
- Conversational AI — be able to hold a conversation with a human

Data Sets

What is a dataset?

A data set is a **logical grouping of units of data** that are closely related and/or share the same data structure.

There are **publicly available data** sets that are used in the **learning of statistics, data analytics, machine learning**

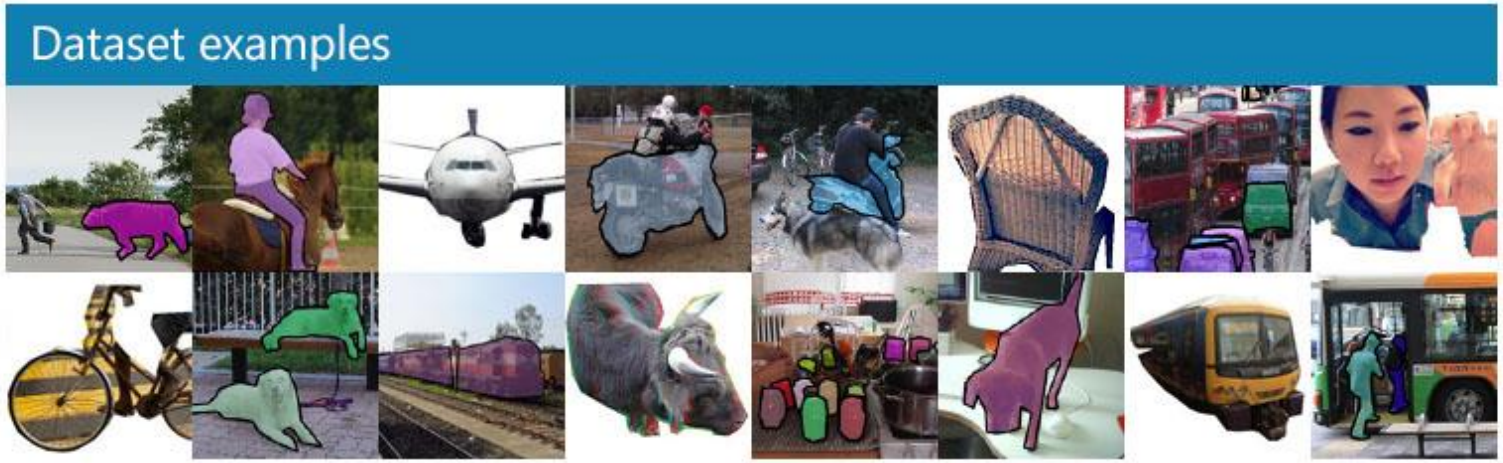
MNIST database

Images of **handwritten digits** used to test classification, clustering, and image processing algorithm.

Commonly used when learning how to build computer vision ML models to translate handwriting into digital text

More Data Sets....

Common Objects In Context (COCO) dataset
A dataset which contains many common images using a JSON file (coco format) that indentify objects or segments within an image.



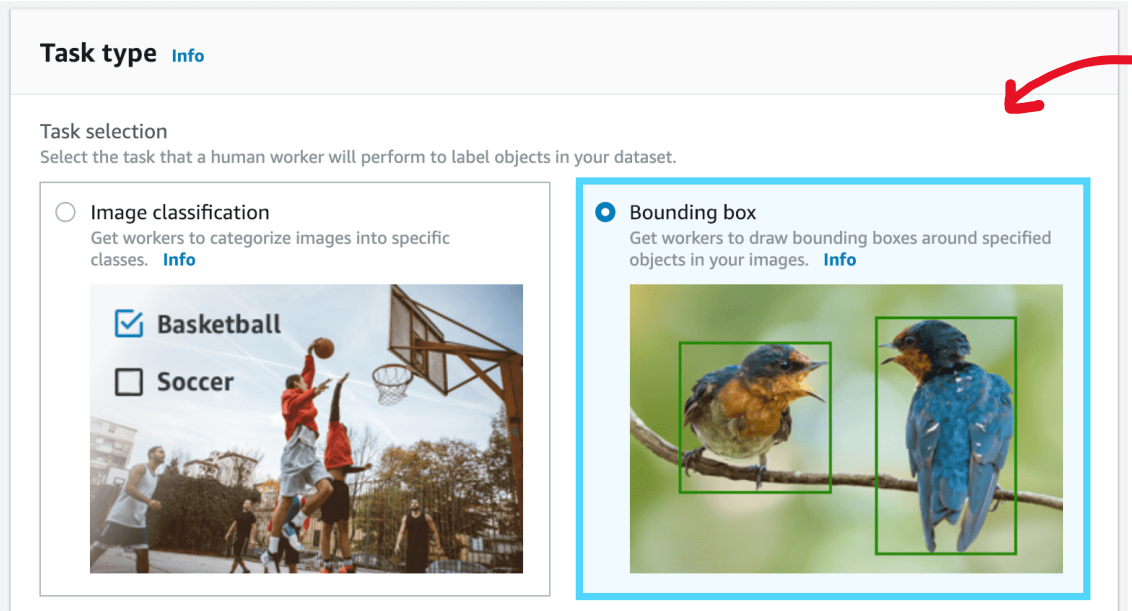
This dataset features:

- Object segmentation
- Recognition in context
- Superpixel stuff segmentation
- 329K images (>200K labeled)
- 0.5 million object instances
- 79 object categories
- 90 stuff categories
- 4 captions per image
- 249,000 people with keypoints

What is Labeling?

What is Data Labeling?

the process of **identifying raw data** (images, text files, videos, etc.) **and adding one or more meaningful and informative labels** to **provide context** so that a machine learning model can learn



With **supervised machine learning**, labeling is a prerequisite to produce training data and **each piece of data will *generally* be labeled by a human**

With **unsupervised machine learning**, **labels will be produced by the machine**, and may not be human readable.

What is a ground truth?

A properly labeled dataset that you use as the objective standard to train and assess a given model is often called "ground truth." The accuracy of your trained model will depend on the accuracy of your ground truth

Supervised vs Unsupervised vs Reinforcement

Supervised Learning (SL)
Data that has been labeled for training
*Task-driven – **make a prediction***

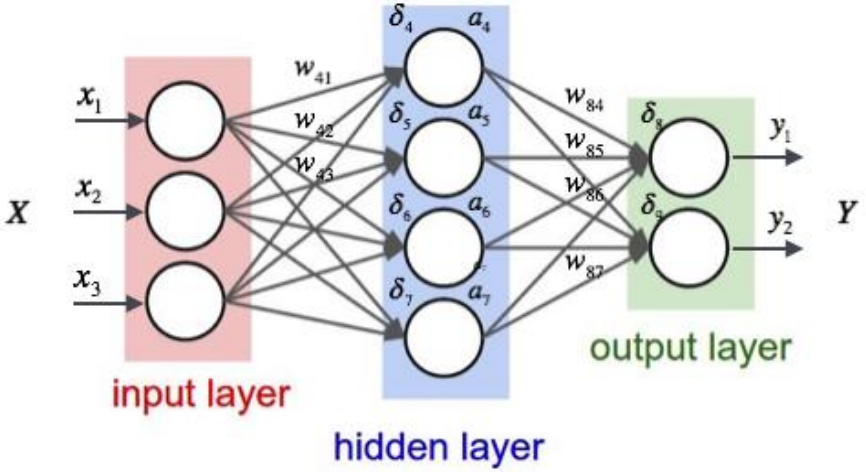
When the labels are known and you want a precise outcome.
When you need a specific value returned
eg. Classification, Regression

Unsupervised Learning (SL)
Data has not been labeled, the ML model needs to do its own labeling
*Data-driven — **recognize a structure or pattern***

When the labels are not known and the outcome does not need to be precise.
When you're trying to make sense of data.
eg. Clustering, Dimensionality Reduction, Association

Reinforcement Learning (RI)
There is no data, there is an environment and an ML model generates data any many attempt to reach a goal
*Decisions-driven — **Game AI, Learning Tasks, Robot Navigation***

Neural Networks and Deep Learning



Activation Functions
An algorithm applied to a hidden layer node that affects connected output e.g. ReLu

Dense
When the next layer increases the amount of nodes

Sparse
When the next layer decreases the amount of nodes

What are Neural Networks? (NN)
Often described as **mimicking the brain**, a **neuron/node represents an algorithm**.
Data is inputted into a neuron and based on the output the data will be passed to one of many other connected neurals.
The connection between neurons is weighted.
The network is organized in layers.
There will be a input layer, 1 to many hidden layers and an output layer.

What is Deep Learning?
A neural network that has **3 or more hidden layers** is considered deep learning.

What is Feed Forward? (FNN)
Neural Networks where connections between nodes do not form a cycle (always move forward)

What is Backpropagation (BP)?
Moves backwards through the neural network adjusting weights to improve outcome on next iteration. This is how a neural net learns.

Loss Function
A function that compares the ground truth to the prediction to determine the error rate (how bad the network performed)

What is a GPU?

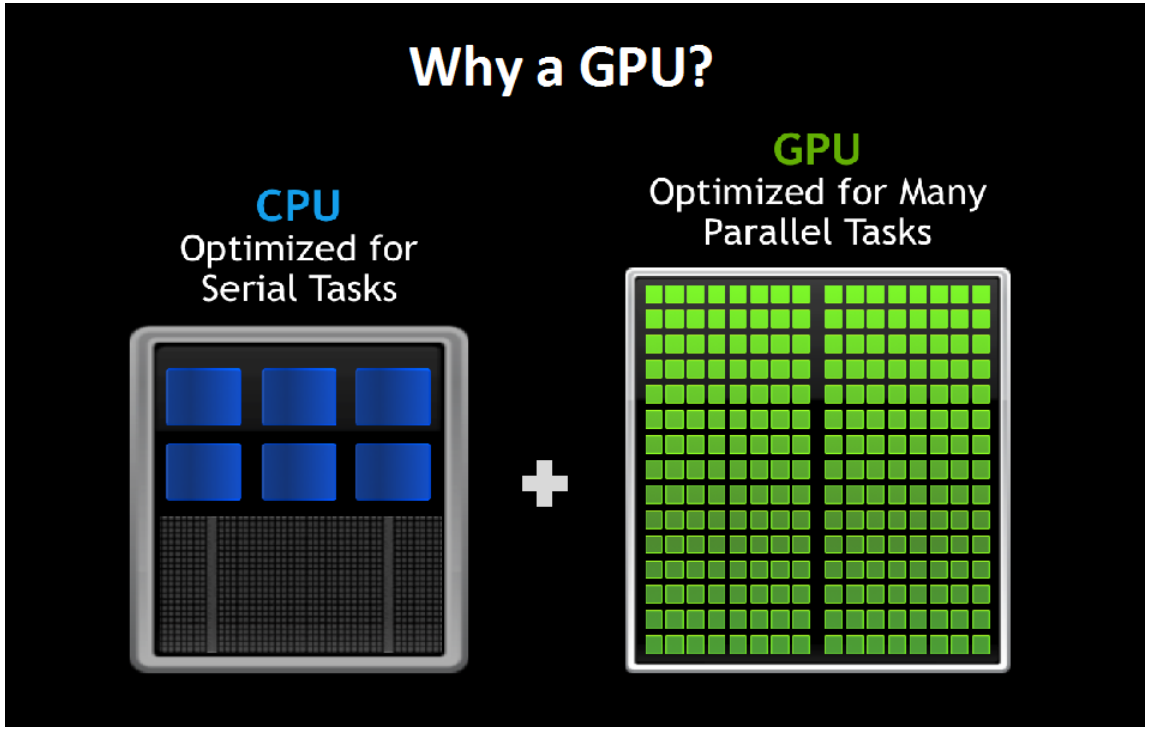
What is a GPU?
A General Processing Unit (GPU) that is specially designed to quickly render high-resolution images and video **concurrently.**

GPUs can perform **parallel operations on multiple sets of data**, and so they are commonly used for **non-graphical tasks such as machine learning** and scientific computation.

CPU can have average 4 to 16 processor cores…
GPUs can **thousands of processor cores**
4 to 8 GPUs can provide as many as 40,000 cores

GPUs are best suited for repetitive and highly-parallel computing tasks:

- Rendering graphics
- Cryptocurrency mining
- Deep Learning and ML



What is CUDA?

What is NVIDA?

NVIDA is a company that manufactures **graphical processing units (GPUs)** for gaming and professional markets

What is CUDA?

Compute Unified Device Architecture (CUDA) is a **parallel computing platform** and **API** by NVIDIA that allows developers to use **CUDA-enabled GPUs** for general-purpose computing on GPUs (GPGPU)

All major deep learning frameworks are integrated with **NVIDIA Deep Learning SDK**

The NVIDA Deep Learning SDK is a collection of NVIDIA libraries for deep learning.
One of those libraries is the **CUDA Deep Neural Network library (cuDNN)**
cuDNN provides highly tuned implementations for standard routines such as:

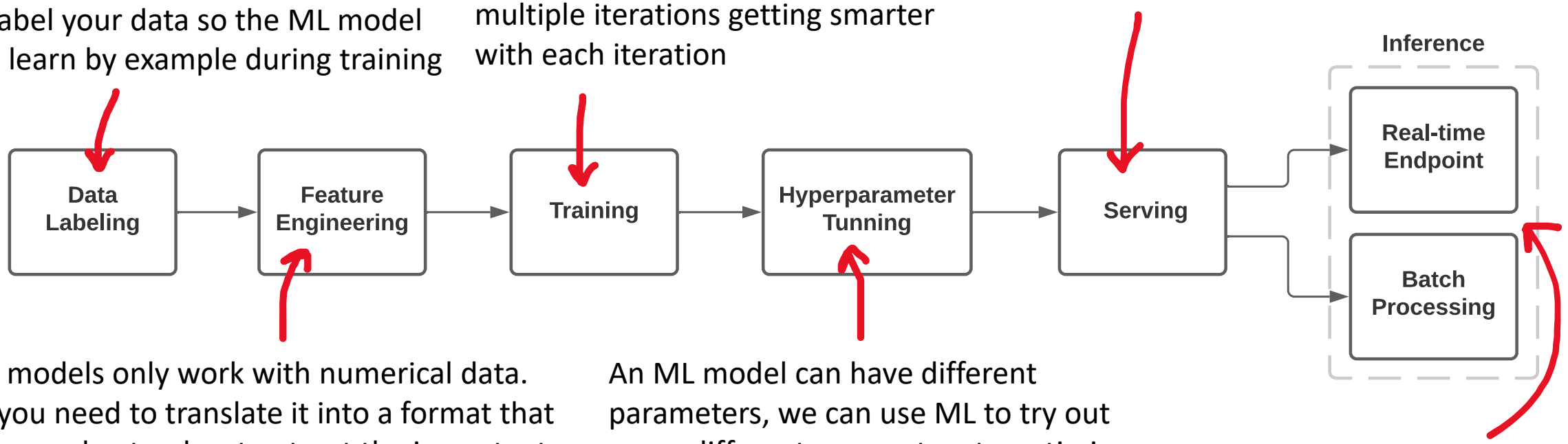
- forward and backward convolution
- Pooling
- Normalization
- activation layers

ML Pipeline

For supervised learning you need to label your data so the ML model can learn by example during training

Your model needs to learn how to become smarter. It will perform multiple iterations getting smarter with each iteration

We need to make are ML model accessible, so we serve by hosting in a virtual machine or container.



ML models only work with numerical data. So you need to translate it into a format that it can understand, extract out the important data that the ML needs to focus on.

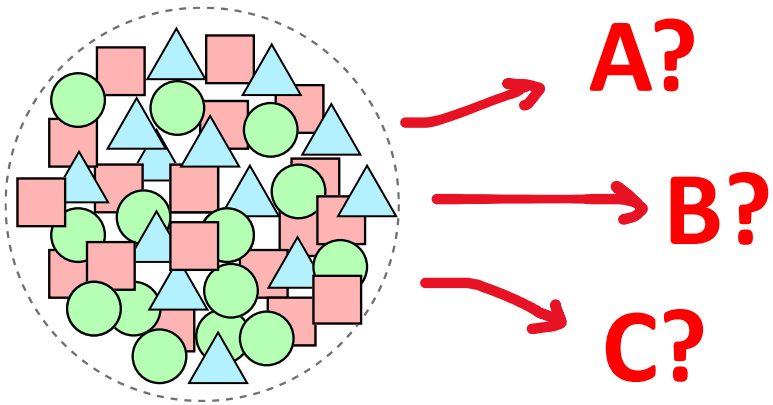
An ML model can have different parameters, we can use ML to try out many different parameters to optimize the outcome

Inference is the act of requesting to make a prediction

Very simplified ML pipeline

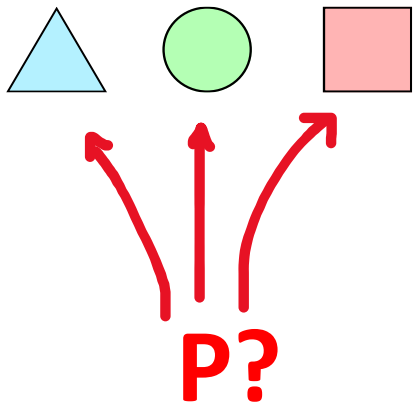
Forecasting vs Prediction

What is a **Forecasting**?



- Makes a future prediction with **relevant data**
  - analysis of trends
  - Its not “guessing”

What is a **Prediction**?



- Makes a future prediction **without relevant data**
  - uses statistics to predict future outcomes
  - Its more of “guessing”
  - Uses decision theory

What are Metrics?

Performance/Evaluation Metrics are used to evaluate different Machine Learning Algorithms

For different types of problems different metrics matter, *(this is not an exhaustive list)*

- Classification Metrics (accuracy, precision, recall, F1-score, ROC, AUC)
- Regression Metrics (MSE, RMSE MAE)
- Ranking Metrics (MRR, DCG, NDCG)
- Statistical Metrics (Correlation)
- Computer Vision Metrics (PSNR, SSIM, IoU)
- NLP Metrics (Perplexity, BLEU, METEOR, ROUGE)
- Deep Learning Related Metrics (Inception score, Frechet Inception distance)

There are two categories of evaluation metrics

- Internal Evaluation — metrics used to evaluate the internals of the ML model
  - Accuracy, F1 Score, Precision, Recall (The Famous Four) used in all kinds of models
- External Evaluation — metrics used to evaluate the final prediction of the ML model

What is Jupyter Notebook?

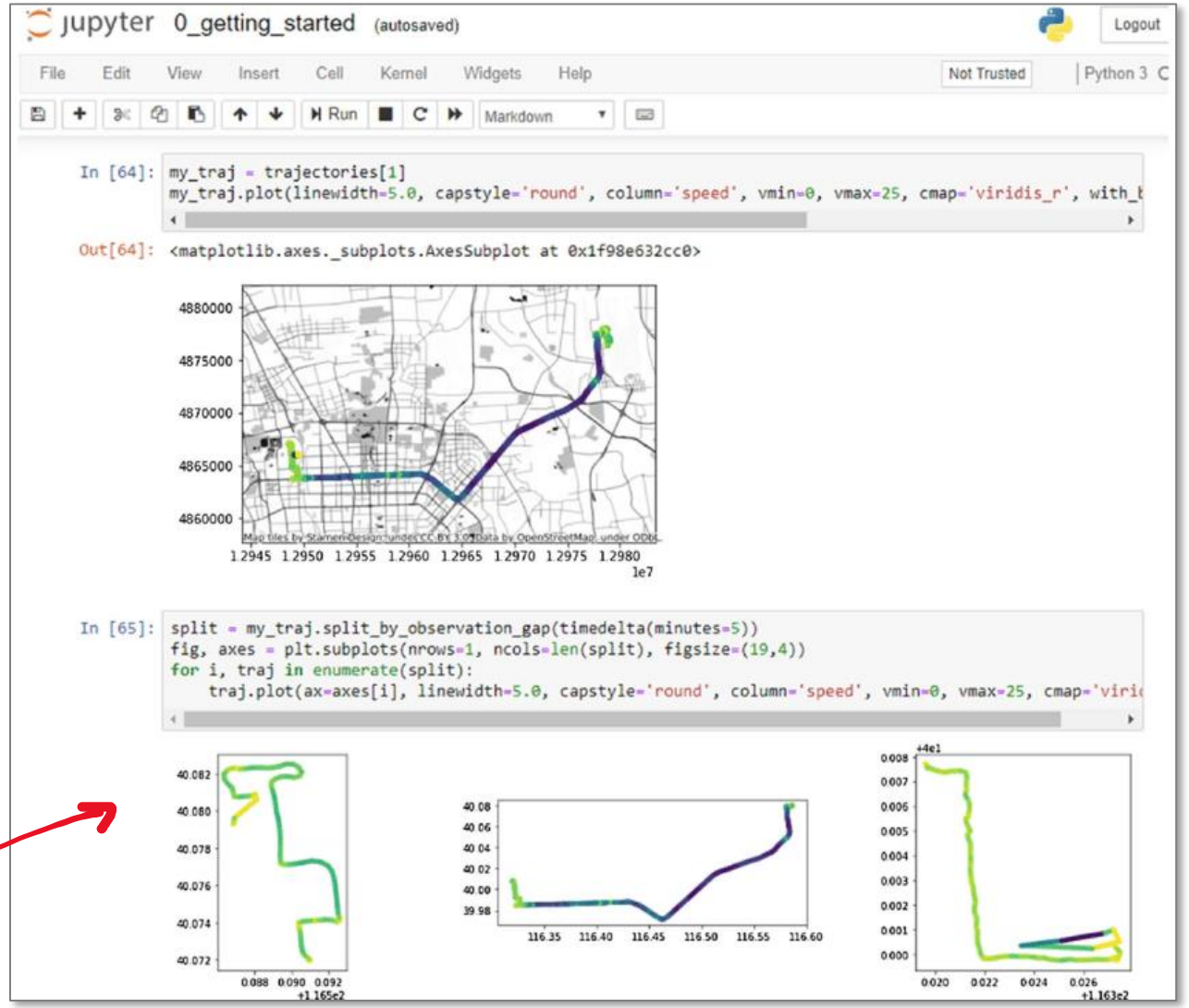
Jupyter Notebook
A Web-based application for authoring documents that combine:

- live-code
- narrative text
- equations
- visualizations

iPython's notebook feature *became* Jupyter Notebook

Jupyter Notebooks were overhauled and better integrated into an IDE called **JupyterLab**

You generally want to open Notebooks in Labs
The legacy web-based interface is known as
Jupyter *classic* notebook



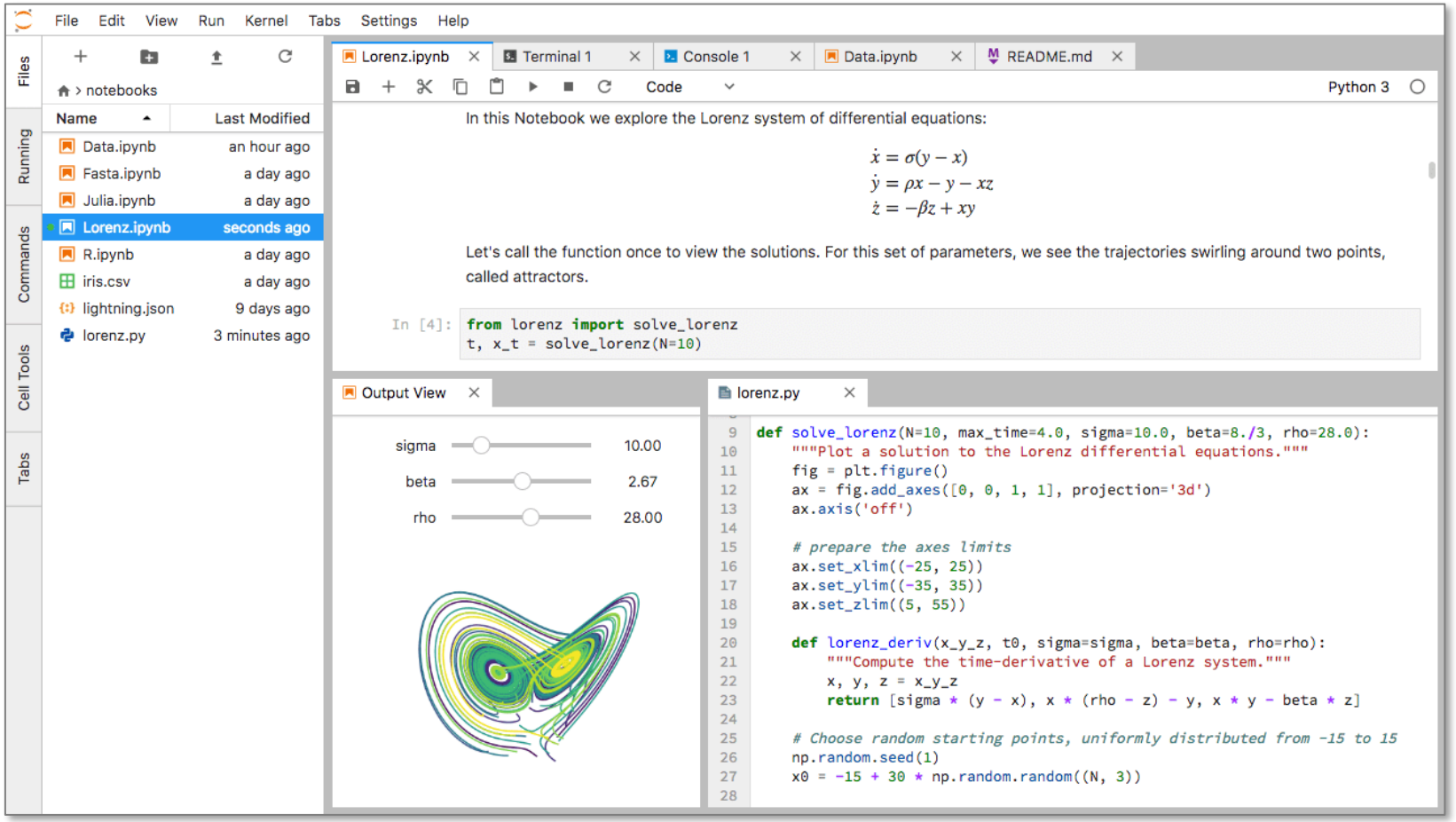
What is JupyterLab?

JupyterLab is a a ***next-generation* web-based user interface**

All the familiar features of the classic Jupyter Notebook in a flexible and powerful user interface:

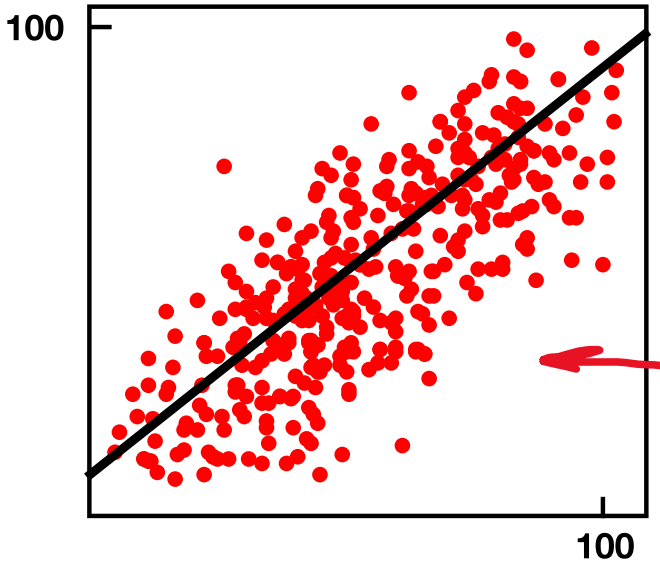
- notebook
- Terminal
- text editor
- file browser
- rich outputs

JupyterLab will *eventually replace* the classic Jupyter Notebook



Regression

Regression is a process of finding a function to **correlate a labeled dataset into continuous variable/number**.

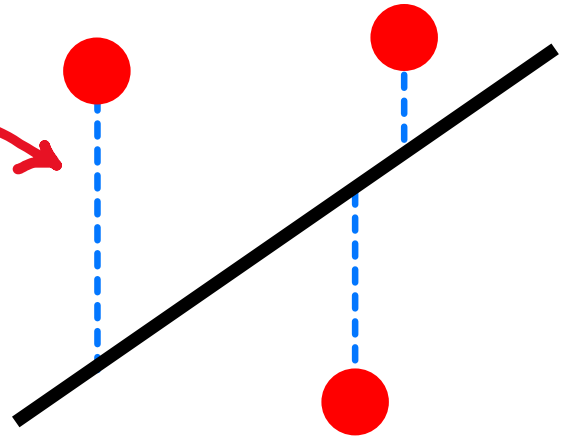


Outcome: Predict this **variable** in the future

What will the temperature be next week?
eg. **20°C**

Vectors (dots) are plotted on a graph in multiple dimensions eg (X,Y)
A regression line is drawn though the dataset.

The **distance** of the vector from the regression line called an **Error**



Different **Regression algorithms** use the error to predict future variables:

- Mean squared error (MSE)
- Root mean squared error (RMSE)
- Mean absolute error (MAE)

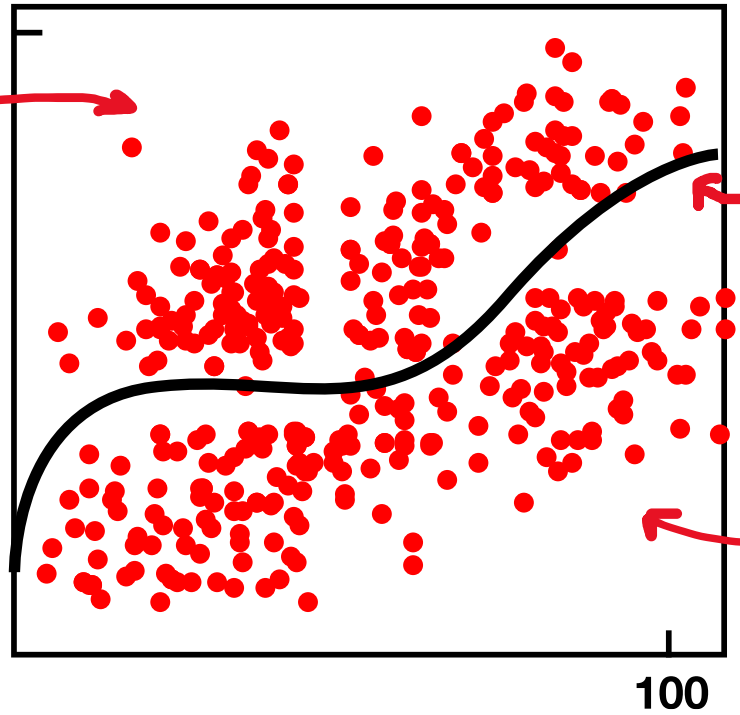
Classification

Classification is a process of finding a function to **divide a labeled dataset into classes/categories**

Outcome: Predict **category** to apply to the inputted data

Will it rain next Saturday?

eg. **Sunny, Rainy**



Data over here will be **Sunny**

A **classification line** divides the dataset

Data over here will be **Rainy**

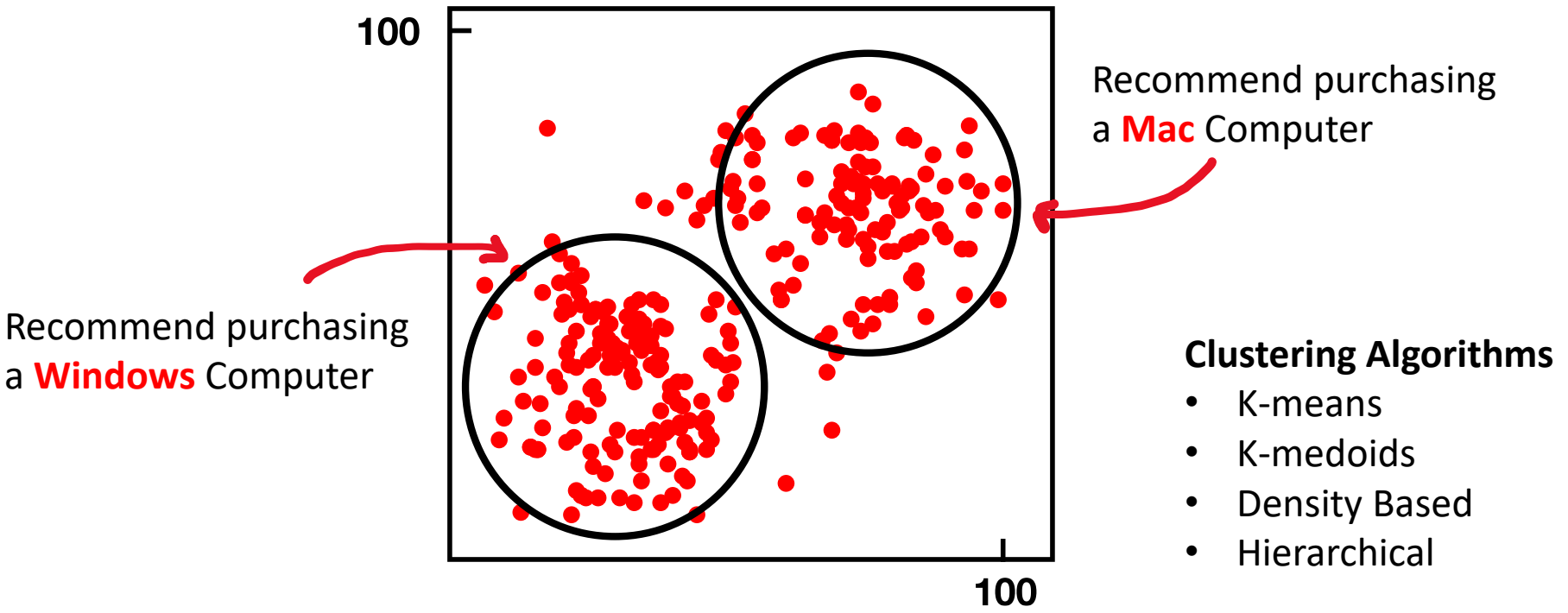
Classification Algorithms

- Logistic Regression
- Decision Tree/Random Forest
- Neural Networks
- Naive Bayes
- K-Nearest Neighbors
- Support Vector Machines

Clustering

Clustering is a process **grouping unlabeled data based on similarities and differences**.

Outcome: Group data based on their similarities or differences



Clustering Algorithms

- K-means
- K-medoids
- Density Based
- Hierarchical

Classification Metrics – Confusion Matrix

A confusion matrix is table to visualize the **model predictions** (predicted) vs **ground truth labels** **(actual)**
Also known as an error matrix. They are useful in classification problems

How many people ate the banana?

	Predicted **NO** negative	Predicted **YES** positive
Actual **NO** False	75 True Negatives (TN)	25 False Positives (FP)
Actual **YES** true	50 False Negatives (FN)	20 True Positives (TP)

Our ground truth had 100 labeled items
Total False (tF)

Our model made 70 predictions
Total True (tT)

125 were NO
Total Negative (tN)

45 were YES
Total Positive (tP)

The size of matrix is dependent on the labels:
Apple, Banana, Orange 3x2 = 6 cells

We have total 170 items
Total (t)

Anomaly Detection AI

What is an anomaly?
An abnormal thing; a marked deviation from the norm or a standard

What is anomaly detection?
Anomaly Detection is the process of finding outliers within a dataset called an **anomaly**
Detecting when a piece of data or access patterns appear suspicious or malicious

Use cases for anomaly detection

- Data cleaning
- **Intrusion detection**
- **Fraud detection**
- Systems health monitoring
- Event detection in sensor networks
- Ecosystem disturbances
- Detection of critical and cascading flaws

Anomaly detection by hand is a very tedious process.
Using machine learning for anomaly detection is more efficient and accurate

Anomaly detector Detect anomalies in data to quickly identify and troubleshoot issues.

Computer Vision

Computer Vision is when we use Machine Learning Neural Networks to **gain high-level understanding from digital images or video**

Computer Vision Deep Learning Algorithms:

- **Convolutional neural network (CNN)** — image and video recognition
  - Inspired after how human eyes actually process information and send it back to brain to be processed
- **Recurrent neural network** (**RNN**) — handwriting recognition or speech recognition

Types of Computer Vision

- **Image Classification** — look at an image or video and classify (place it in a category)
- **Object Detection** — identify objects within an image or video and apply labels and location boundaries
- **Semantic Segmentation** — identify segments or objects by drawing pixel mask (great for objects in movement)
- **Image Analysis** — analyze an and image or video to apply descriptive and context labels
  - eg. An employee sitting at a Desk in Tokyo
- **Optical Character Recognition** — Find text in images or videos and extract them into digital text for editing
- **Facial Detection** — detect faces in a photo or video, draw a location boundary, label their expression

Computer Vision

Seeing AI is an AI app developed by Microsoft for iOS
Seeing AI uses the device camera to **identify people and objects**, and then the app audibly **describes those objects for people with visual impairment**.

Azure's Computer Vision Service Offering:

Computer Vision analyze images and video, and extract descriptions, tags, objects, and text

Custom Vision custom image classification and object detection models using your own images

Face Detect and identify people and emotions in images.

Form Recogniser translate scanned documents into key /value or tabular editable data

Natural Language Processing (NLP)

Natural Language Processing is Machine Learning that can **understand the context of a corpus (a body of related text).**

NLP enables you to:

- Analyze and interpret text within documents, email messages
- Interpret or contextualise spoken token e.g., sentiment analysis
- Synthesize speech e.g., a voice assistance talking to you
- Automatically translate spoken or written phrases and sentences between languages.
- Interpret spoken or written commands and determine appropriate actions.

Cortana is a **virtual assistant** developed by Microsoft which uses the Bing search engine to perform tasks such as setting reminders and answering questions for the user.

Natural Language Processing (NLP)

Azure's NLP Service Offering:

Text Analytics

- sentiment analysis to find out what customers think
- Find topic-relevant phrases using key phrase extraction
- identify the language of the text with language detection
- Detect and categorize entities in your text with named entity recognition

Translator

- real-time text translation
- multi-language support

Speech

- transcribe audible speech into readable, searchable text

Language Understanding (LUIS)

- natural language processing service that enables you to understand human language in your own application, website, chatbot, IoT device, and more

Conversational AI

Conversational AI is technology that can **participate in conversations with humans**.

- Chatbots
- Voice Assistants
- Interactive Voice Recognition Systems (IVRS)

Use Cases

- **Online Customer Support** — replaces human agents for replying about customer FAQs, shipping
- **Accessibility** — voice operated UI for those who are visually impaired
- **HR processes** — employee training, onboarding, updating employee information
- **Health Care** — accessible and affordable health care eg. claim processes
- **Internet of Things (IoT)** — Amazon Alexa, Apple Siri and Google Home
- **Computer Software** — autocomplete search on phone or desktop

QnA Maker Create a conversational question-and-answer bot from your existing content (Knowledge base).

Azure Bot Service Intelligent, serverless bot service that scales on demand. Used for creating, publishing, and managing bots

Responsible AI

Responsible AI focuses on **ethical, transparent and accountable** use of AI technologies

Microsoft puts into practice Responsible AI via its six **Microsoft AI principles**

1. **Fairness —** AI systems should treat all people fairly
2. **Reliability and Safety —** AI systems should perform reliably and safely
3. **Privacy and Security —** AI systems should be secure and respect privacy
4. **Inclusiveness —** AI systems should empower everyone and engage people
5. **Transparency —** AI systems should be understandable
6. **Accountability —** People should be accountable for AI systems

Responsible AI – Fairness

AI systems should treat all people fairly

AI systems can reinforce existing societal stereotypical
Bias can be introduced during the development of a pipeline

AI systems that are used to allocate or withhold:

- opportunities
- resources
- Information

In domains:

- Criminal Justice
- Employment and Hiring
- Finance and Credit

eg. an ML model designed to select final applicants for a hiring pipeline without incorporating any bias based on gender, ethnicity or may result in an unfair advantage

Azure ML can tell you how each feature can influence a model's prediction for bias

Fairlearn

Fairlearn is an open-source python project to help data scientist to improve fairness in their AI systems

Responsible AI – Reliability and safety

AI systems should perform reliably and safely

AI software must be **rigorous tested** to ensure they work as expected before release to the end user

If there are scenarios where AI is making mistakes its important to release a report **quantified risks and harms** to end-users so they are informed of the short-comings of an AI solution

AI where concern for reliability and safety for humans is critically important:

- Autonomous Vehicle
- AI health diagnosis, AI suggesting prescriptions
- **Autonomous Weapon Systems**

Responsible AI – Privacy and security

AI systems should be secure and respect privacy

AI can require vasts amounts of data to train Deep Learning ML models.
The nature of the ML model may require **Personally identifiable information (PII)**

It is important that we ensure protection of user data that it is not leaked or disclosed

In some cases ML Models can be run locally on a user's device so their PII remains on their device avoiding that vulnerability

AI Security Principles to detect malicious actors:

- Data Origin and Lineage
- Data Use Internal vs External
- Data Corruption Considerations
- Anomaly detection

Responsible AI – Inclusiveness

AI systems should empower everyone and engage people

If we can design AI solutions for the **minority** of users
Then we can design AI solutions for the majority of users

Minority Groups

- physical ability
- gender
- sexual orientation
- ethnicity
- other factors

Responsible AI – Transparency

AI systems should be understandable

Interpretability / Intelligibility is when end-users can understand the behaviour of the UI

Transparency of AI systems can result in

- Mitigating unfairness
- Help developers debug their AI systems
- Gaining more trust from our users

Those build AI systems should be:

- open about the why they are using AI
- open about the limitations of their AI systems

Adopting an open-source AI framework can provide transparency (at least from a technical perceptive) on the internal workings of an AI systems

Responsible AI – Accountability

People should be accountable for AI systems

The structure put in place to consistently enacting AI principles and taking them into account

AI systems should work within:

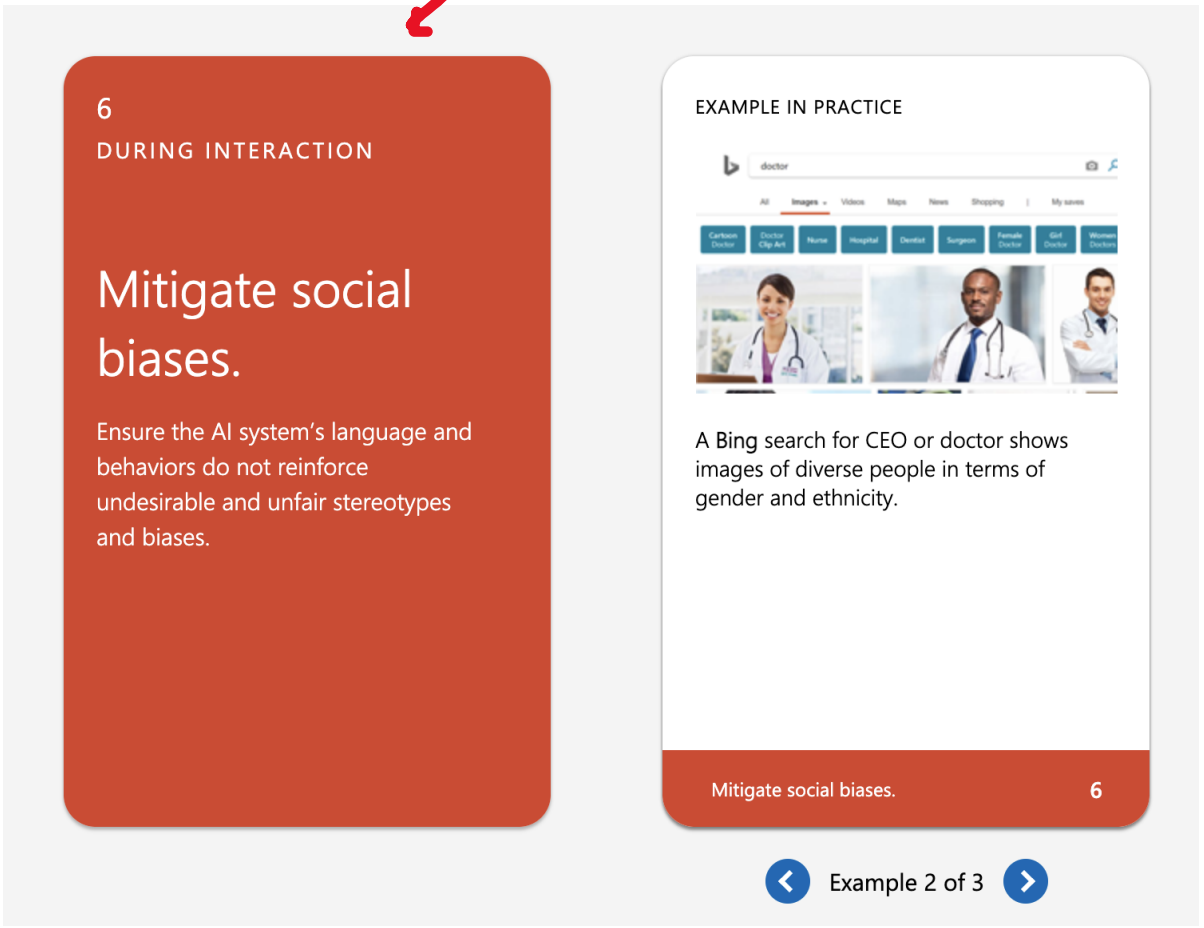
- framework of governance
- organizational principles

ethical and legal standards that are clearly defined

Principles guide Microsoft on how they **Develop, Sell and Advocate** when working with third-parties and this can push towards regulations towards AI Principles

Guidelines for Human-AI Interaction

Microsoft has a free web-app that goes through **practical scenarios** to teach Microsoft AI Principles



https://aidemos.microsoft.com/guidelines-for-human-ai-interaction/demo

Azure Cognitive Services

Azure Cognitive Services is a **comprehensive family of AI services** and cognitive APIs to help you build intelligent apps

Create customizable, pretrained models built with *"breakthrough"* AI research

Deploy Cognitive Services anywhere from the cloud to the edge with containers

Get started quickly—no machine-learning expertise required

Developed with strict ethical standards, empowering responsible use with industry-leading tools and guidelines

Azure Cognitive Services

Decision

- **Anomaly Detector** — Identify potential problems early on.
- **Content Moderator —** Detect potentially offensive or unwanted content.
- **Personaliser —** Create rich, personalized experiences for every user.

Language

- **Language Understanding** — Build natural language understanding into apps, bots and IoT devices.
- **QnA Maker** — Create a conversational question and answer layer over your data.
- **Text Analytics** — Detect sentiment, key phrases and named entities.
- **Translator** — Detect and translate more than 90 supported languages.

Speech

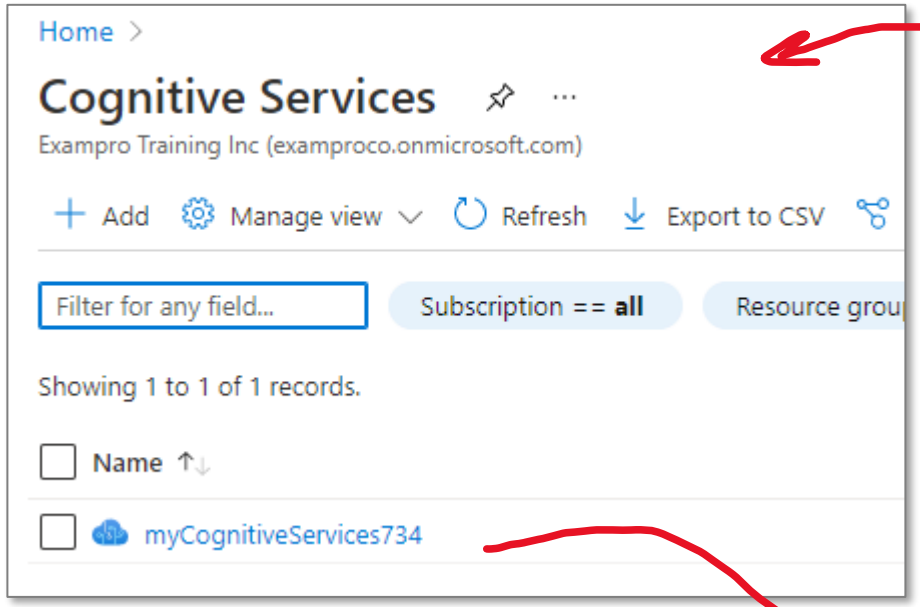
- **Speech to Text** — Transcribe audible speech into readable, searchable text.
- **Text to Speech** — Convert text to lifelike speech for more natural interfaces.
- **Speech Translation** — Integrate real-time speech translation into your apps.
- **Speaker Recognition** — Identify and verify the people speaking based on audio.

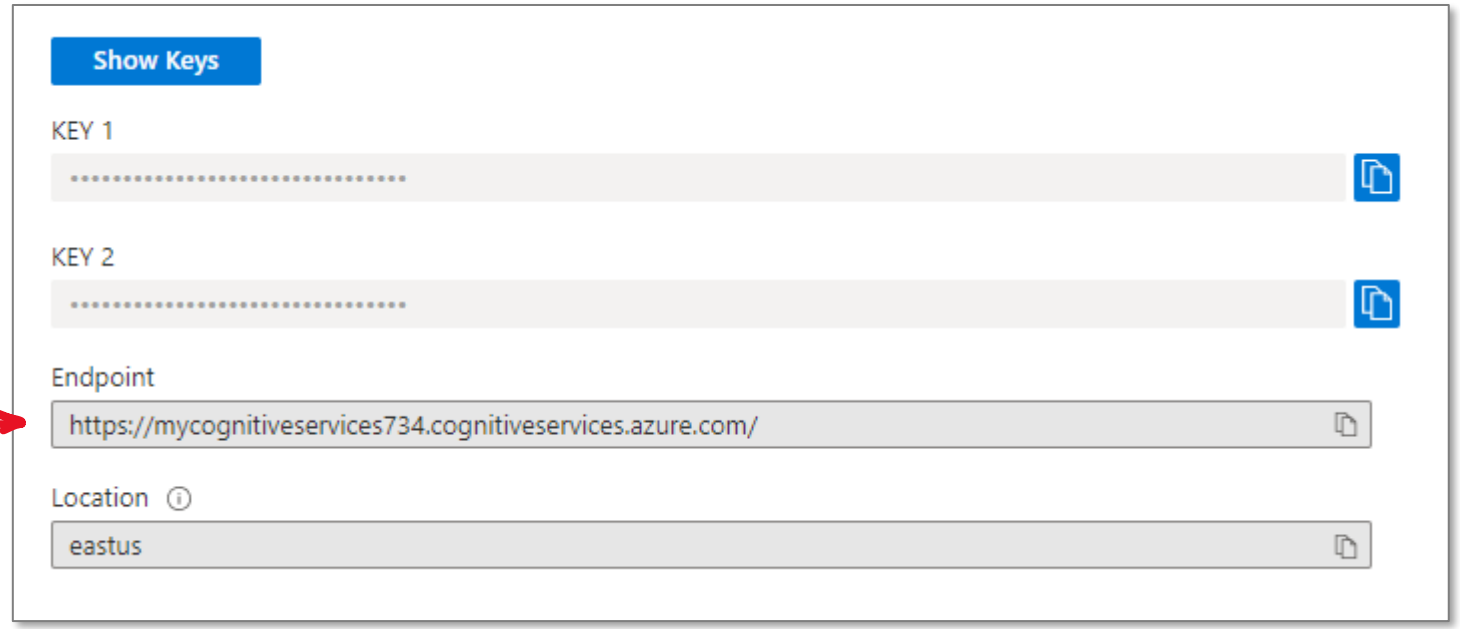
Vision

- **Computer Vision** — Analyze content in images and video.
- **Custom Vision** — Customize image recognition to fit your business needs.
- **Face** — Detect and identify people and emotions in images.

Azure Cognitive Services

Cognitive Services is an umbrella AI service that enables customers to **access multiple AI services** with an **API key and an API Endpoint**





Knowledge Mining

Knowledge mining is a **discipline** in AI that uses a **combination of intelligent services to quickly learn from vast amounts of information.**

It allows organizations to deeply understand and easily explore information, uncover hidden insights, and find relationships and patterns at scale.

Ingest

Structured
- Databases
- CSV

Unstructured
- PDF, Video, Image, Audio

Ingest content from a range of sources, using connectors to first and third-party data stores.

Enrich

Cognitive Services
- Vision
- Language
- Speech
- Decision
- Search

Enrich the content with AI capabilities that let you extract information, find patterns, and deepen understanding.

Explore

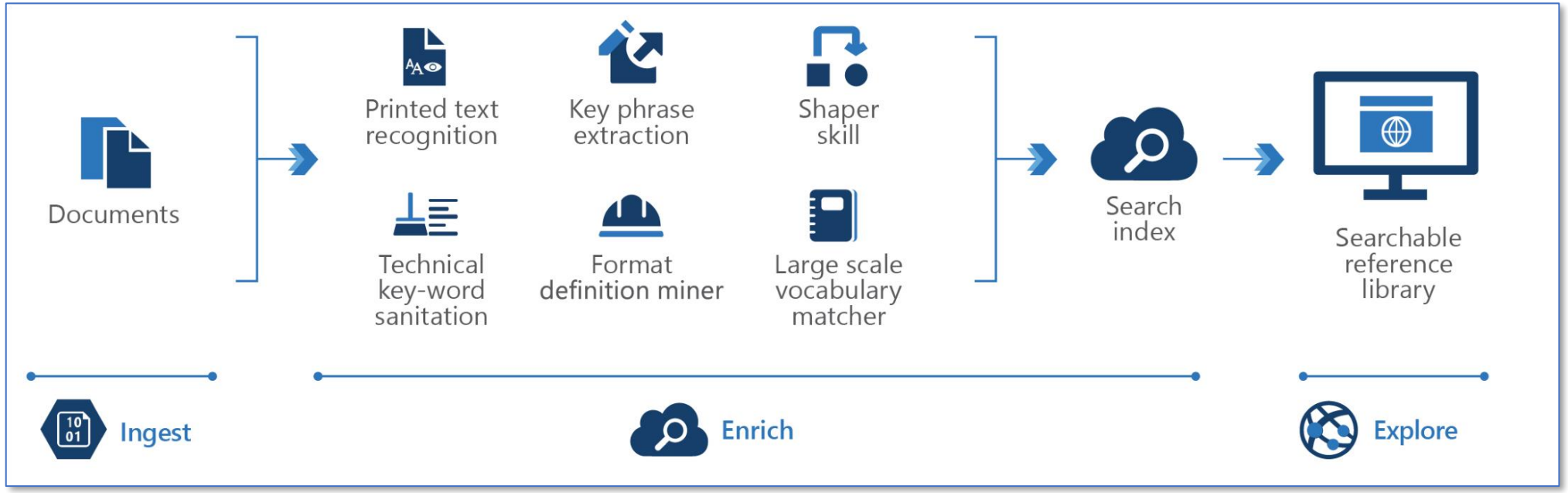
Enrich, Structured Data
- Customer relationship management (CRM)
- RAP Systems
- Power BI

Explore the newly indexed data via search, bots, existing business applications, and data visualizations.

Knowledge Mining – Use Cases

Content research

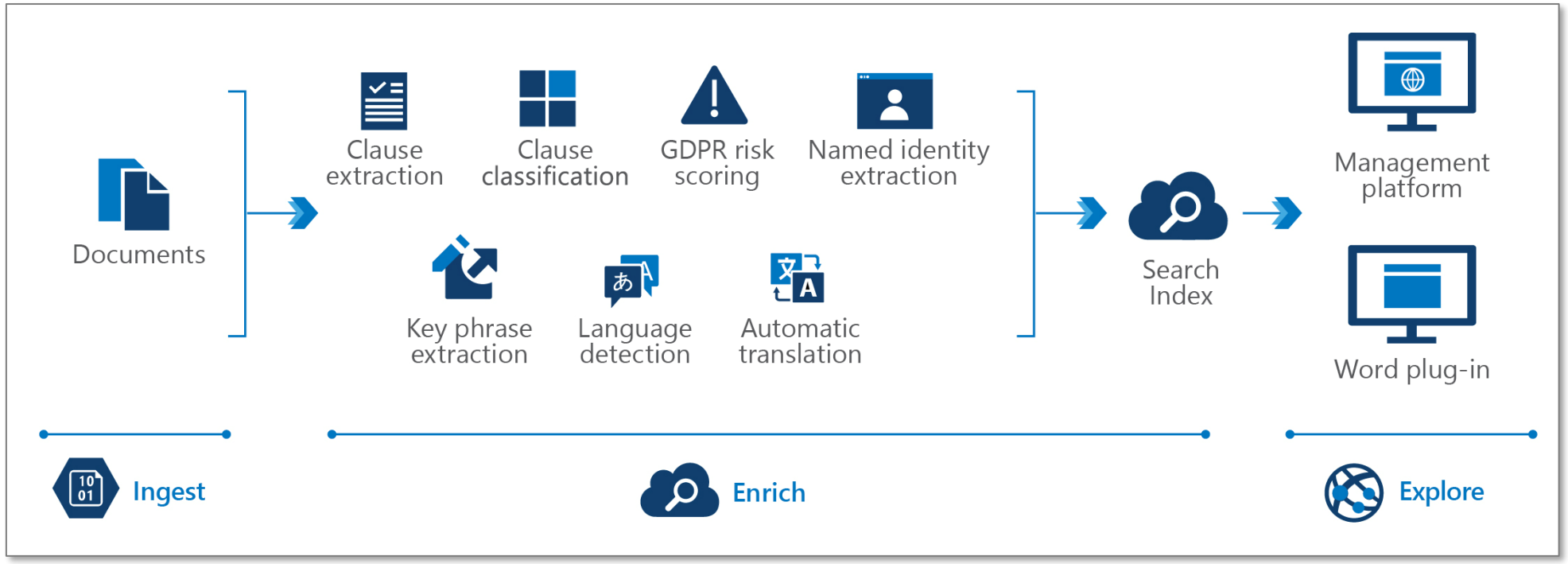
When organizations task employees to review and research of technical data, it can be tedious to read page after page of dense text. Knowledge mining helps employees quickly review these dense materials.



Knowledge Mining – Use Cases

Auditing, risk, and compliance management

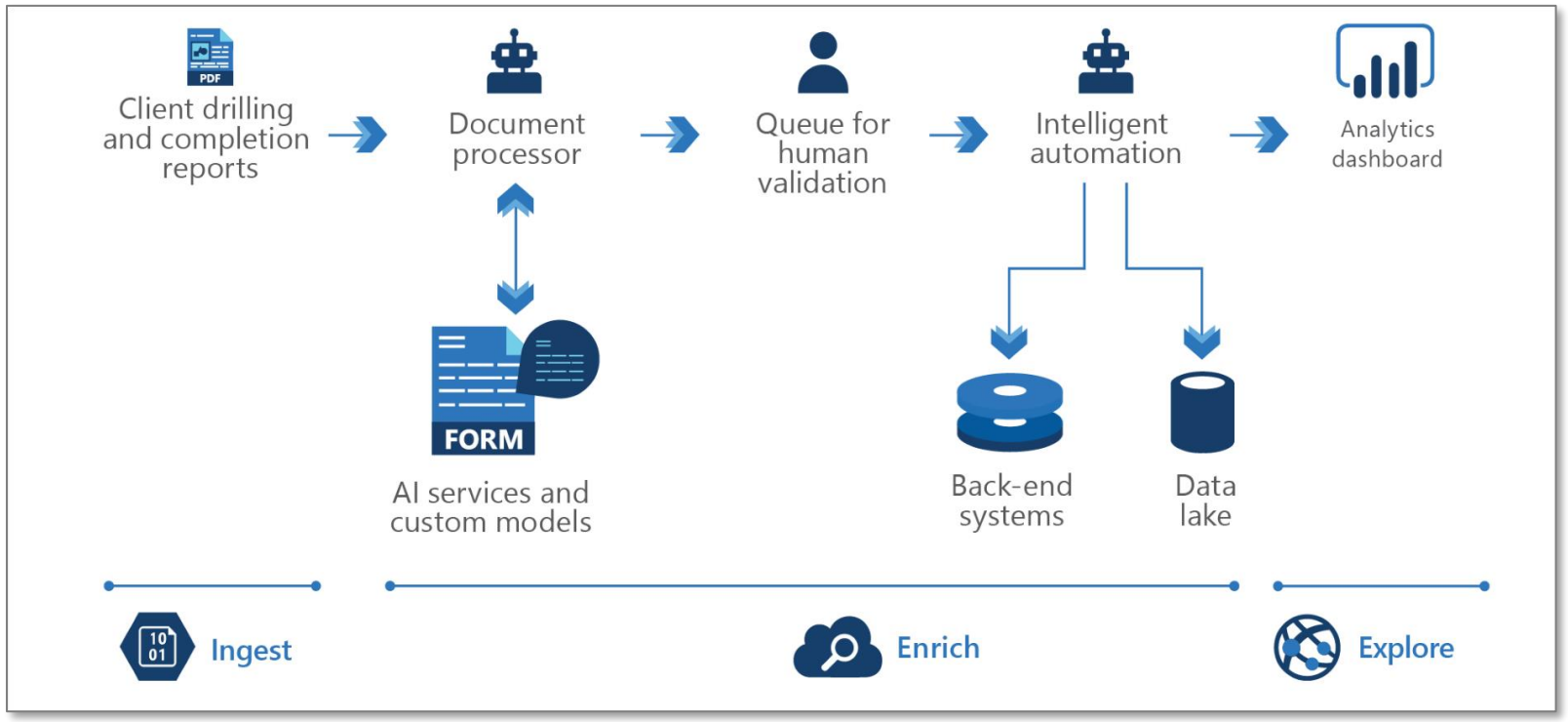
Developers could use knowledge mining to help attorneys quickly identify entities of importance from discovery documents and flag important ideas across documents.



Knowledge Mining – Use Cases

Business process management

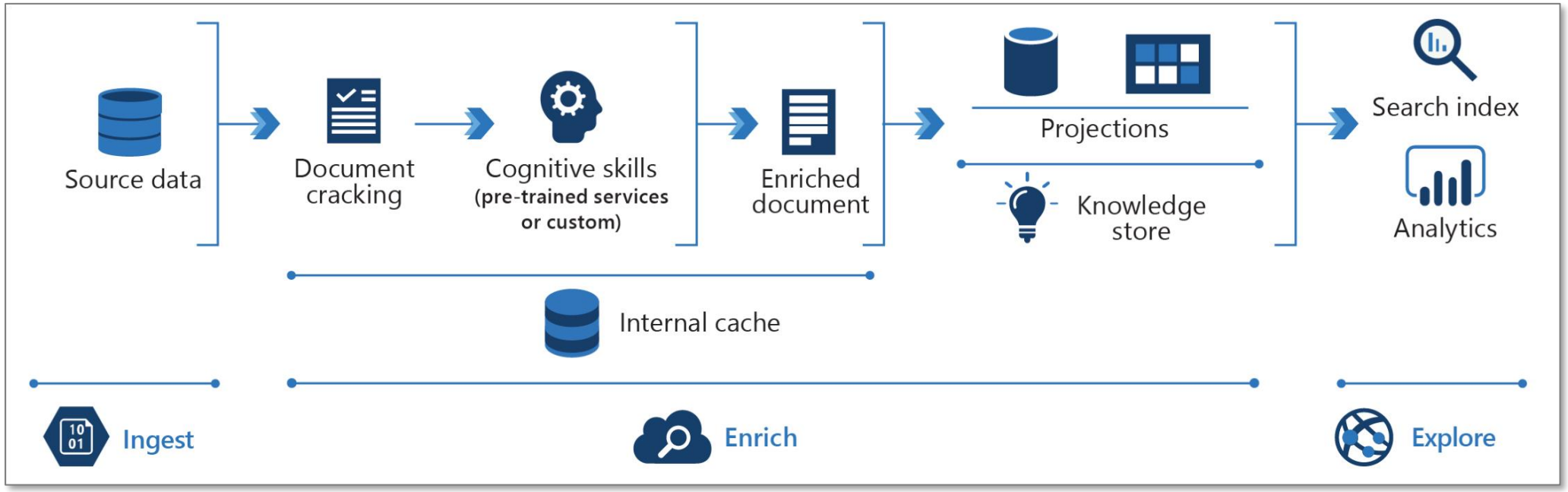
In industries where bidding competition is fierce, or when the diagnosis of a problem must be quick or in near real-time, companies can use knowledge mining to avoid costly mistakes.



Knowledge Mining – Use Cases

Customer support and feedback analysis

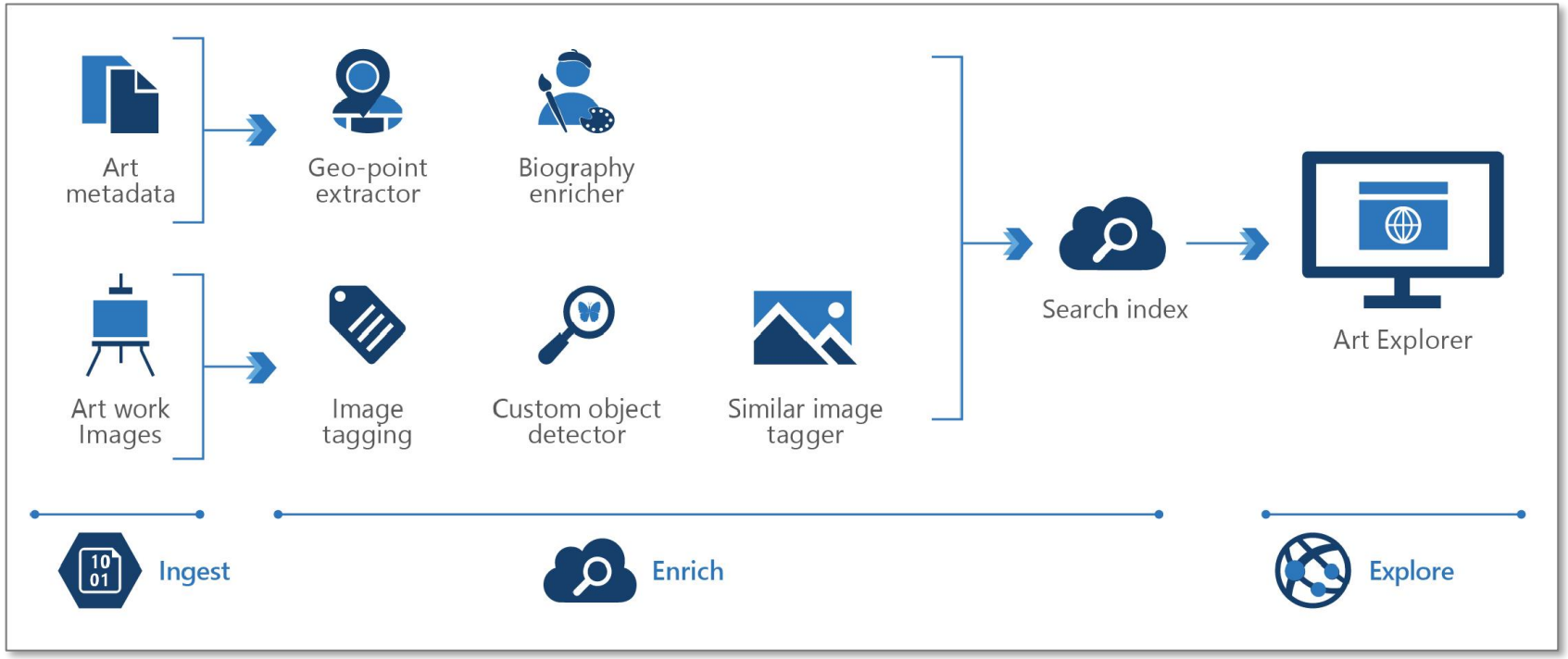
For many companies, customer support is costly and inefficient. Knowledge mining can help customer support teams quickly find the right answer for a customer inquiry or assess customer sentiment at scale.



Knowledge Mining – Use Cases

Digital asset management

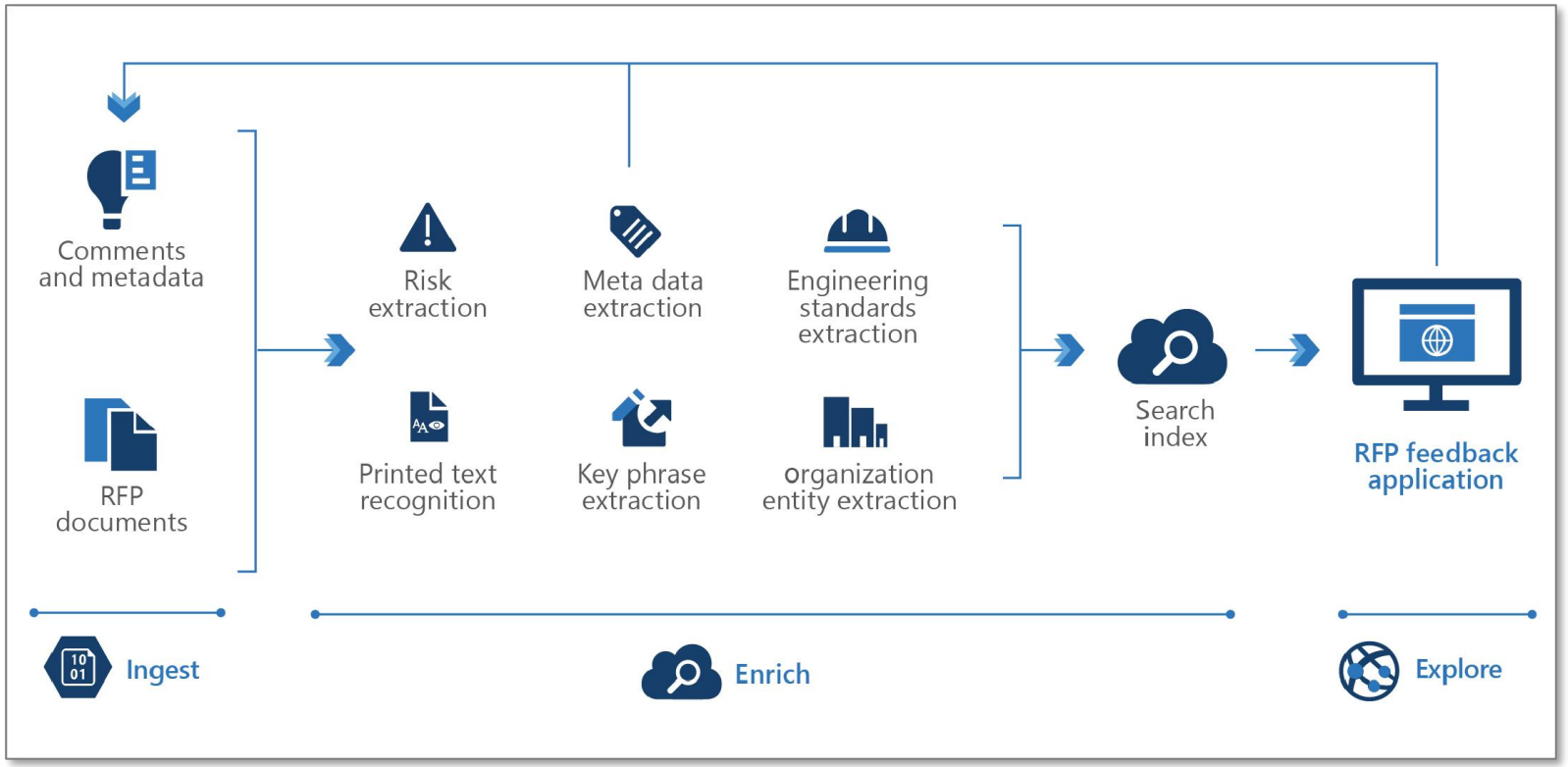
Given the amount of unstructured data created daily, many companies are struggling to make use of or find information within their files. Knowledge mining through a search index makes it easy for end customers and employees to locate what they are looking for faster.



Knowledge Mining – Use Cases

Contract management

Many companies create products for multiple sectors, hence the business opportunities with different vendors and buyers increases exponentially. Knowledge mining can help organizations to scour thousands of pages of sources to create an accurate bid.



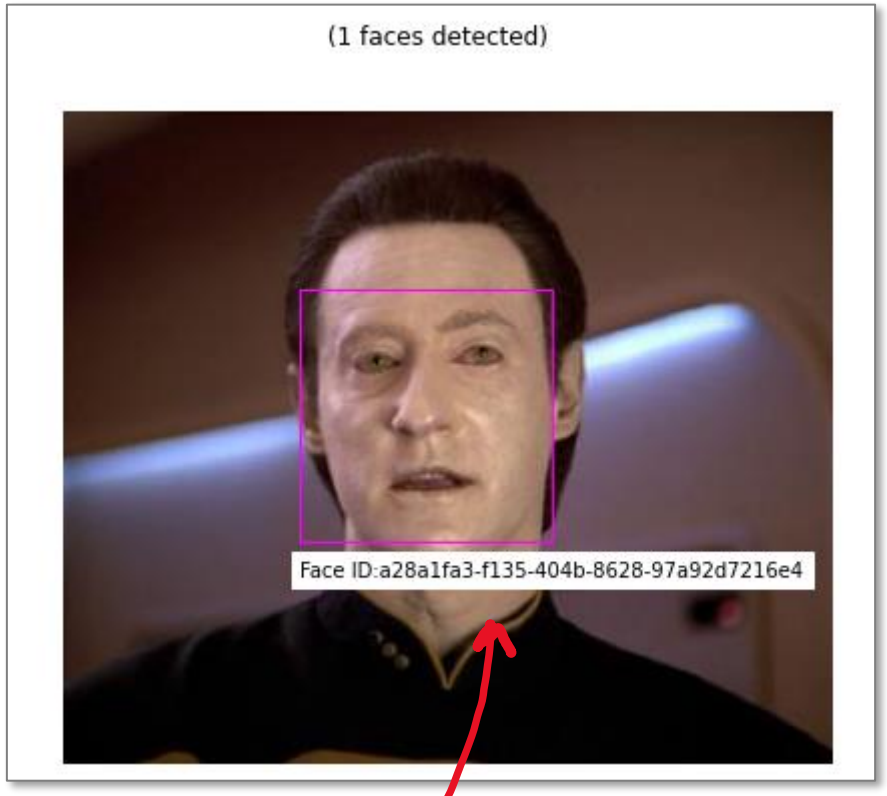
Face Service

Azure Face service provides AI algorithms that **detect, recognize, and analyze human faces** in images

Azure Face can detect:

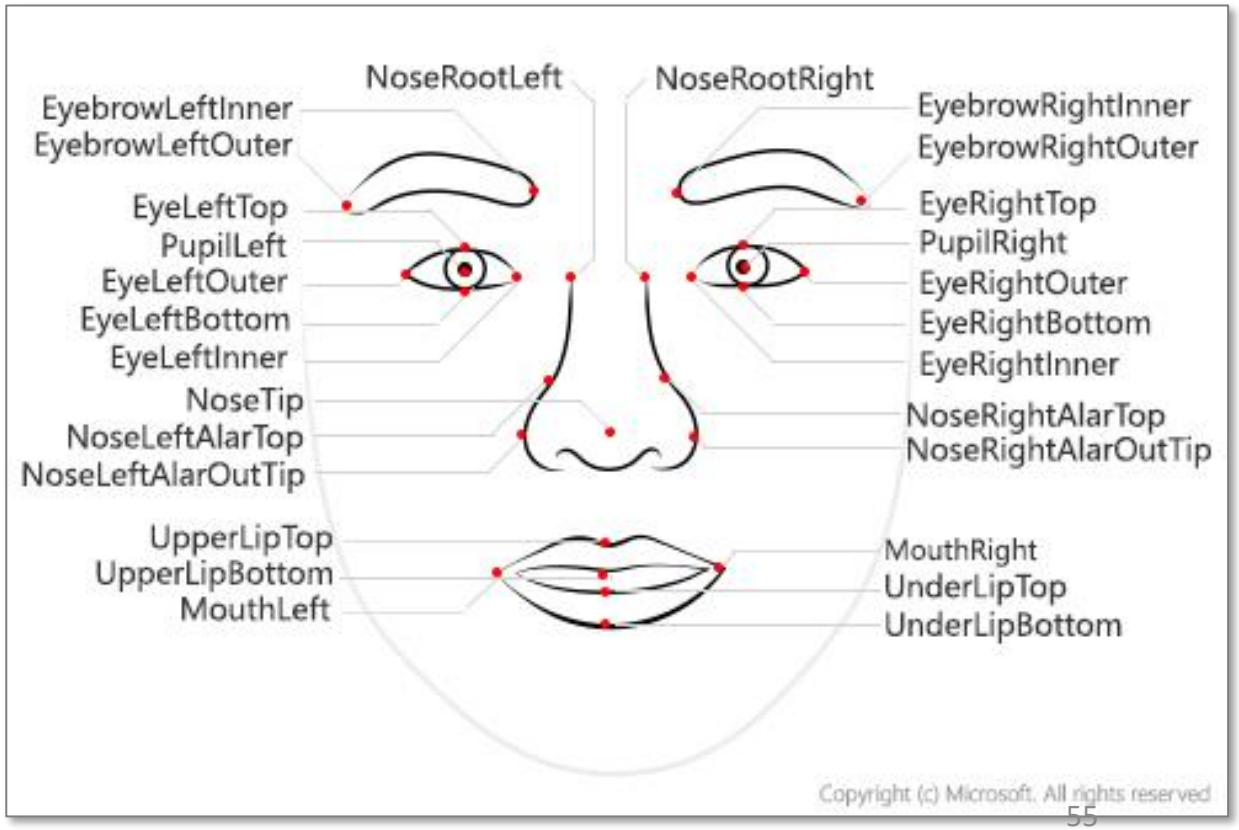
- faces in an image
- faces with specific attributes
- face landmarks
- similar faces
- the same face as a specific identity across a gallery of images

Face Service



Face ID
unique identifier string for each detected face in an image

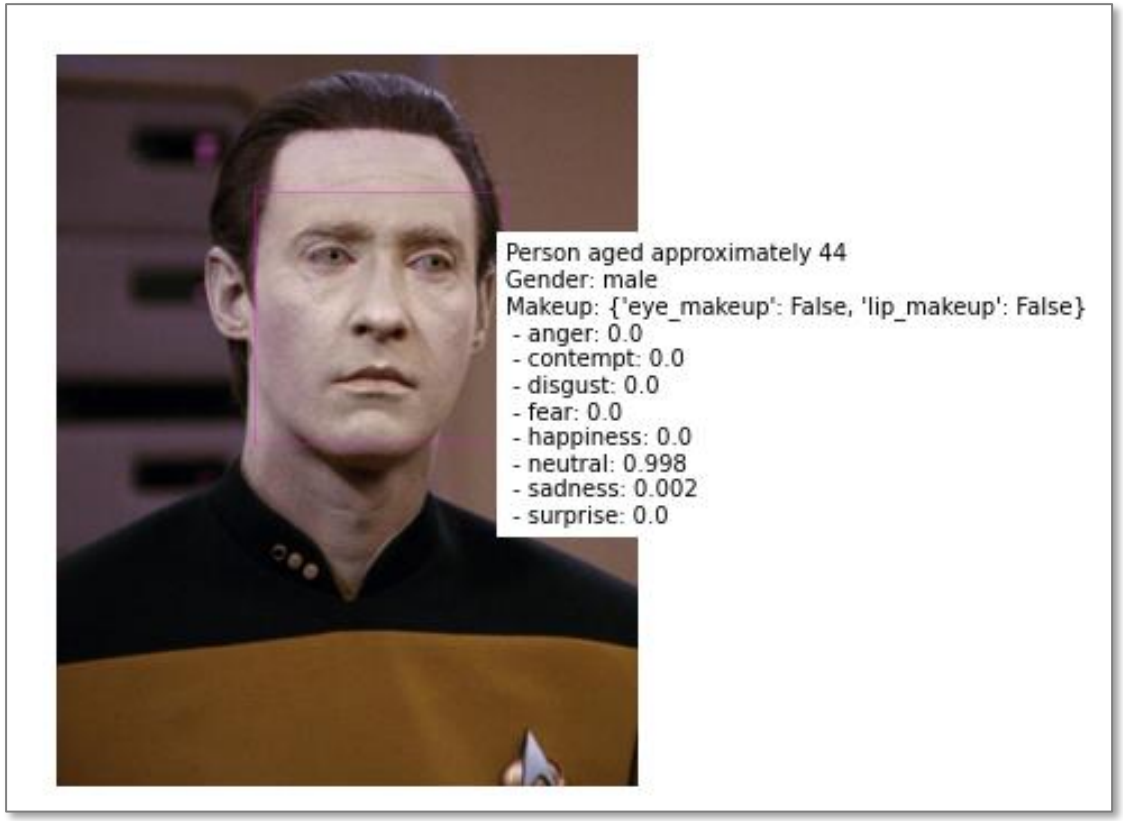
Face Landmarks
easy-to-find points on a face
27 predefined landmark points.



Face Service

Face Attributes

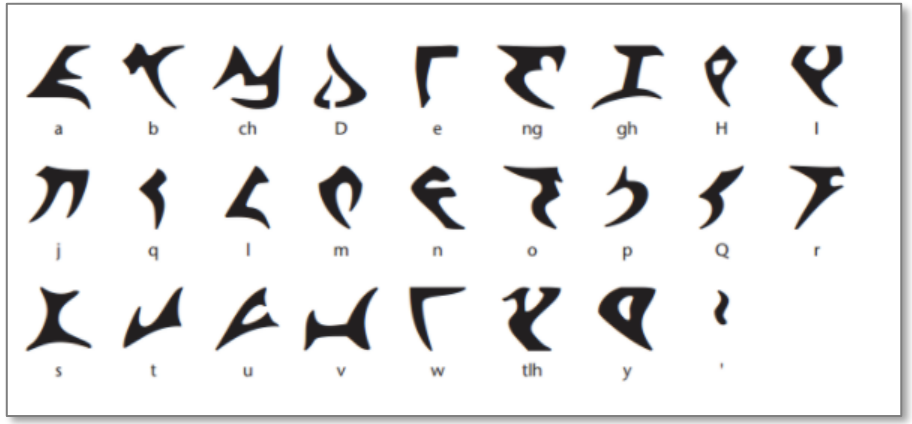
- **Accessories**. (Wearing accessories)
- **Age**
- **Blur** (blurriness of the face in the image)
- **Emotion**.
- **Exposure**
- **Facial hair**
- **Gender**
- **Glasses**
- **Hair**
- **Head pose**
- **Makeup**
- **Mask**. (are they wearing a mask?)
- **Noise**. The visual noise detected in the face image
- **Occlusion**. (objects blocking parts of the face)
- **Smile**



Speech and Translate Service

Azure's Translate service is a **translation service**.

- It can translate 90 languages and dialects
  - It even supports **Klingon!**
- It uses Neural Machine Translation (NMT) replacing its legacy Statistical Machine Translation (SMT)
- **Custom Translator** allows you to extend the service for translation based on your business and domain use case



Azure Speech service can **speech synthesis service** speech-to-text, text-to-speech, and speech-translation

Speech-to-Text
- Real-time Speech-to-text
- Batch Speech-to-Text
- Multi-device Conversation
- Conversation Transcription
- Create Custom Speech Models

Text-to-Speech
- using Speech Synthesis Markup Language (SSML)
- Create Custom Voices

Voice Assistance
- integrates with Bot Framework SDK

Speech Recognition
- Speaker verification & identification

Text Analytics

Text Analytics API is a **Natural Language Processing (NLP)** service for **text mining and text analysis**

Text Analytics can perform:

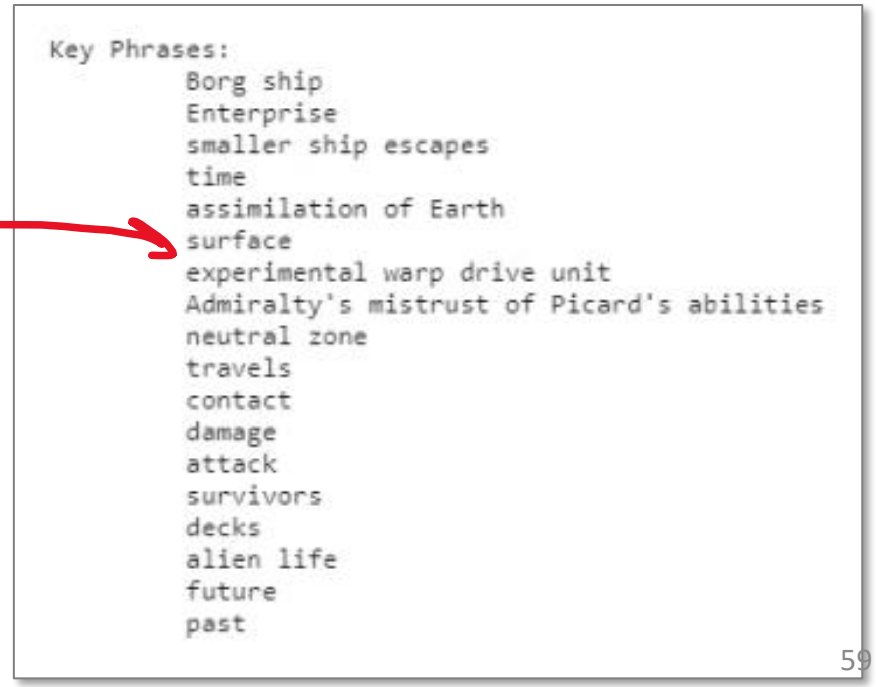
- **sentiment analysis**
  - find out what people think of your brand or topic
    - feature provides sentiment labels (such as "negative", "neutral" and "positive")
- **opinion mining**
  - aspect-based sentiment analysis
  - granular information about the opinions related to aspects
- **key phrase extraction**
  - quickly identify the main concepts in text.
- **language detection**
  - detect the language an input text is written in
- **named entity recognition (NER)**
  - Identify and categorize entities in your text as people, places, organizations, quantities
  - Subset of NER is Personally Identifiable Information (PII)

NLP – Key Phrase Extraction

Key Phrase Extraction quickly identify the main concepts in text

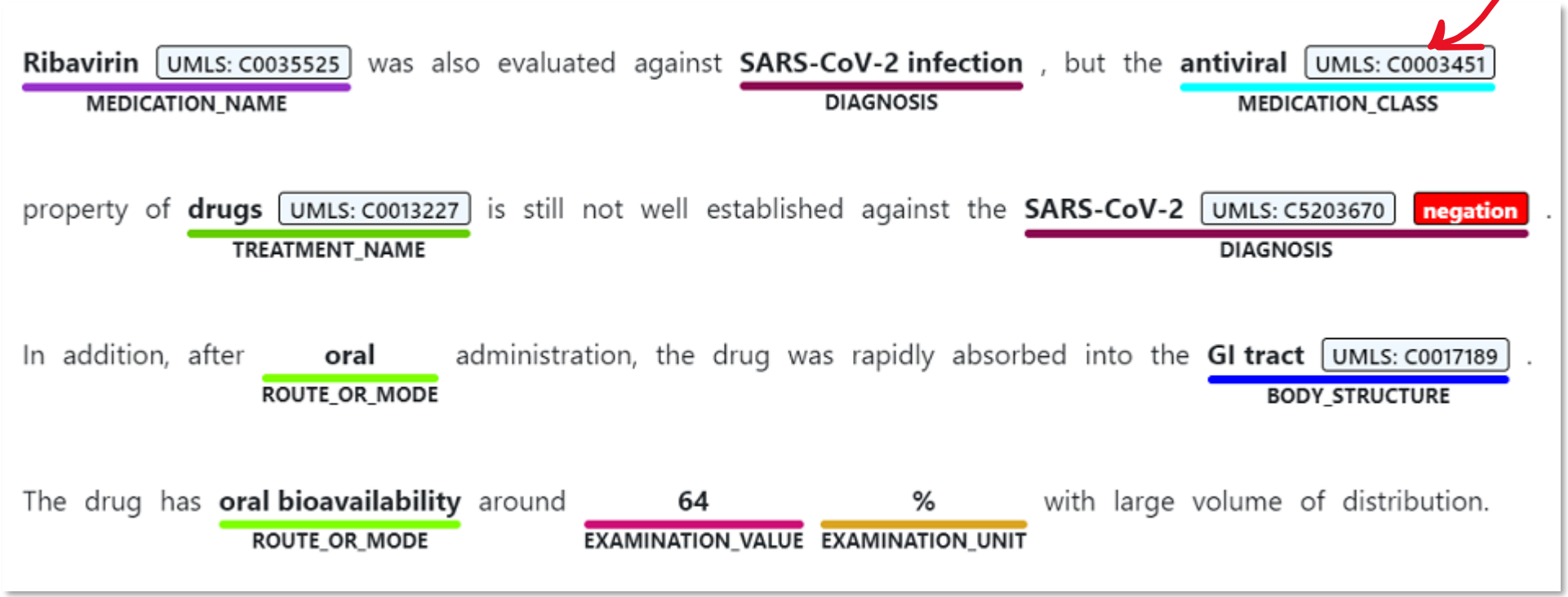
- Key phrase extraction works best when you give it bigger amounts of text to work on
- This is opposite from sentiment analysis, which performs better on smaller amounts of text
- Document size must be 5,120 or fewer characters per document, and you can have up to 1,000 items (IDs) per collection

```
When the Borg launch an attack on Earth, the
Enterprise is sent to the neutral zone due to the
Admiralty's mistrust of Picard's abilities as he had
been assimilated in the past. The Enterprise however,
disobeys and returns to help destroy the Borg ship.
However a smaller ship escapes and travels back in
time, causing the assimilation of Earth in the future.
The Enterprise follow the ship back in time and have
to undo the damage the ship did on the surface to an
experimental warp drive unit that will led Earth to
it's first contact with alien life. Meanwhile, on the
Enterprise, survivors of the Borg ship begin to
assimilate decks within the ship itself….
```



Named Entity Recognition

Named Entity Recognition detects **words and phrases mentioned in unstructured text** that can be **associated with one or more semantic types**.

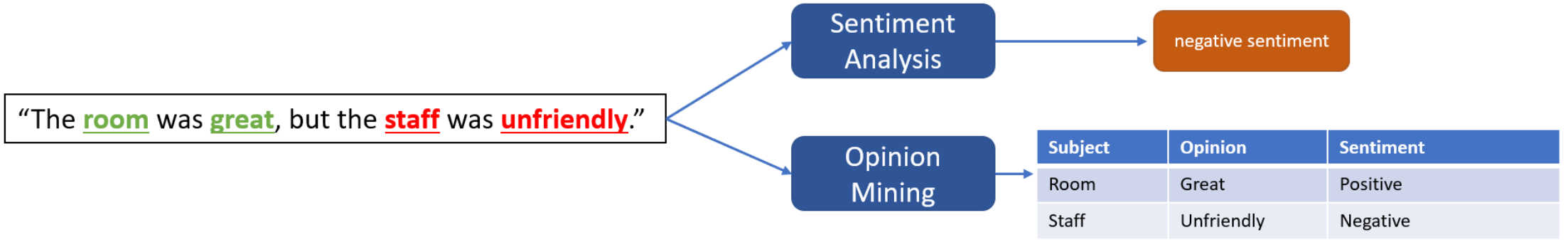


Semantic types could be: *Location, Event, Location, Person, Diagnosis, Age*

NLP – Sentiment Analysis

Sentiment analysis will apply labels and confidence score to text at the **sentence and document level**.

- Labels include **negative, positive, mixed or neutral**
- Confidence scores ranging from 0 to 1



Subject	Opinion	Sentiment
Room	Great	Positive
Staff	Unfriendly	Negative

Opinion mining with provide more granular data with a **Subject** and **Opinion** tied to a Sentient

Language Understanding Service (LUIS)

Language Understanding (LUIS) is **a no-code ML service to build natural language into apps**, bots, and IoT devices.

Quickly create enterprise-ready, custom models that continuously improve.

LUIS is accessed via its own isolate domain at **luis.ai**

LUIS utilizes Natural Language Processing (NLP) and **Natural Language Understanding (NLU)**

NLU is the ability to *transform* a linguistic statement to a representation that enables you to understand your users naturally

LUIS is intended to focus on **intention** and **extraction**:

- What the user wants
- What they are talking about

Language Understanding Service (LUIS)

A LUIS application is composed of a **schema**
This schema is autogenerated for you when you use the LUIS.ai web interface


```
{
  "luis_schema_version": "7.0.0",
  "intents": [
    {
      "name": "None",
      "features": []
    }
  ],
  "entities": [],
  "hierarchicals": [],
  "composites": [],
  "closedLists": [],
  "prebuiltEntities": [],
  "utterances": [],
  "versionId": "0.1",
  "name": "example-app",
  "desc": "",
  "culture": "en-us",
  "tokenizerVersion": "1.0.0",
  "patternAnyEntities": [],
  "regex_entities": [],
  "phraselists": [
  ],
  "regex_features": [],
  "patterns": [],
  "settings": []
}
```


The schema defines:

- **intentions** — what the user is asking for
  - a LUIS app always contains a **None** Intent
- **entities** — what parts of the intent is used to determine the answer
- **utterances** — Examples of user input that includes intent and entities to train the ML model to match predictions against real user input
  - An intent requires one or more example utterance for training
    - It is recommended to have 15-30 example utterances
  - To explicitly train to ignore an utterance use the None Intent

Intents classify user utterances
Entities extract data from utterance

QnA Maker Service

QnA Maker is a **cloud-based Natural Language Processing (NLP) service** that allows you **to create a natural conversational layer** over your data.

QnAMaker is hosted on its own isolate domain at **www.qnamaker.ai**

It will find the most appropriate answer for any input from your **custom knowledge base** (KB) of information

Commonly used to build conversational client applications, which include:

- social media applications
- chat bots
- speech-enabled desktop applications

QnA Maker doesn't store customer data
All customer data is stored in the region the customer deploys the dependent service instances in

QnA Maker Service – Use Cases

When you have static information
Use QnA Maker when you have static information in your knowledge base of answers. This knowledge base is custom to your needs, which you've built with documents such as PDFs and URLs.

When you want to provide the same answer to a request, question, or command
when different users submit the same question, the same answer is returned.

When you want to filter static information based on meta-information
add metadata tags to provide additional filtering options relevant to your client application's users and the information. Common metadata information includes chit-chat, content type or format, content purpose, and content freshness.

When you want to manage a bot conversation that includes static information
your knowledge base takes a user's conversational text or command and answers it. If the answer is part of a pre-determined conversation flow, represented in your knowledge base with multi-turn context, the bot can easily provide this flow.

QnA Maker Service – Knowledgebase

QnA Maker imports your content into a knowledge base of question and answer pairs.

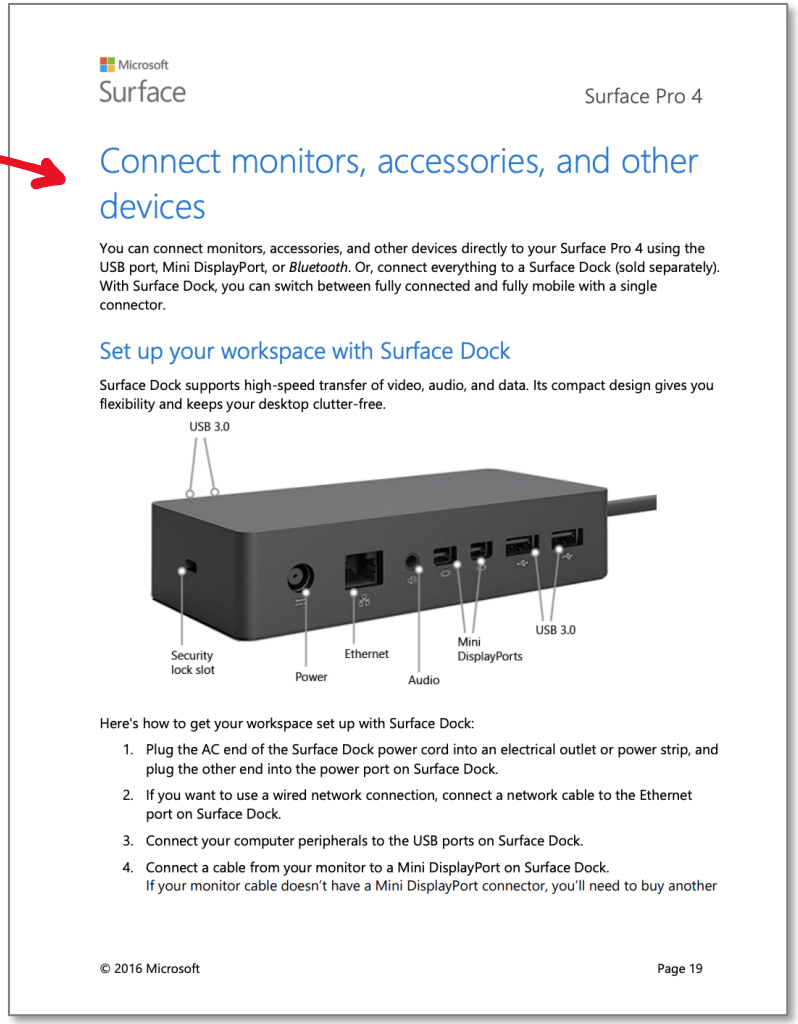
QnA Maker can build you knowledge base from an **existing document, manual or website (URL, DOCX, PDF)**

It will use ML to extract the question and answer pairs.

The content of the question and answer pair includes:

- All the alternate forms of the question
- Metadata tags used to filter answer choices during the search
- Follow-up prompts to continue the search refinement

QnA Maker stores answer text as **markdown**



Microsoft Surface

Surface Pro 4

Connect monitors, accessories, and other devices

You can connect monitors, accessories, and other devices directly to your Surface Pro 4 using the USB port, Mini DisplayPort, or *Bluetooth*. Or, connect everything to a Surface Dock (sold separately). With Surface Dock, you can switch between fully connected and fully mobile with a single connector.

Set up your workspace with Surface Dock

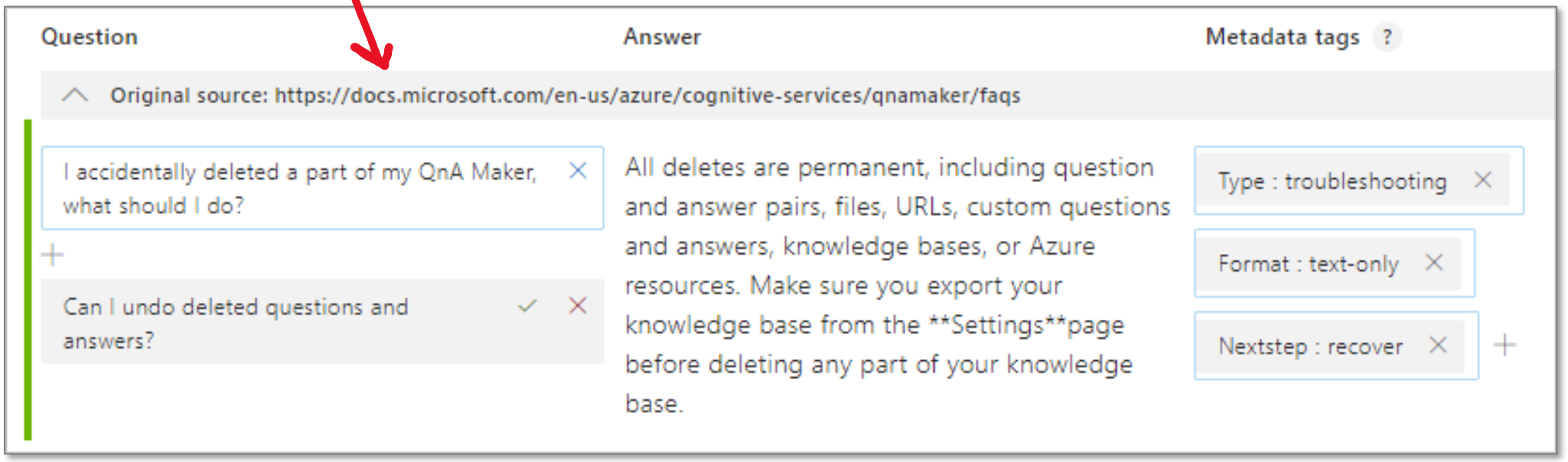
Surface Dock supports high-speed transfer of video, audio, and data. Its compact design gives you flexibility and keeps your desktop clutter-free.

Here's how to get your workspace set up with Surface Dock:

1. Plug the AC end of the Surface Dock power cord into an electrical outlet or power strip, and plug the other end into the power port on Surface Dock.
2. If you want to use a wired network connection, connect a network cable to the Ethernet port on Surface Dock.
3. Connect your computer peripherals to the USB ports on Surface Dock.
4. Connect a cable from your monitor to a Mini DisplayPort on Surface Dock.
   If your monitor cable doesn't have a Mini DisplayPort connector, you'll need to buy another

QnA Maker Service – Knowledgebase

Once your Knowledge Base is imported you can **fine-tune the imported results** by editing the Question and Answer pairs

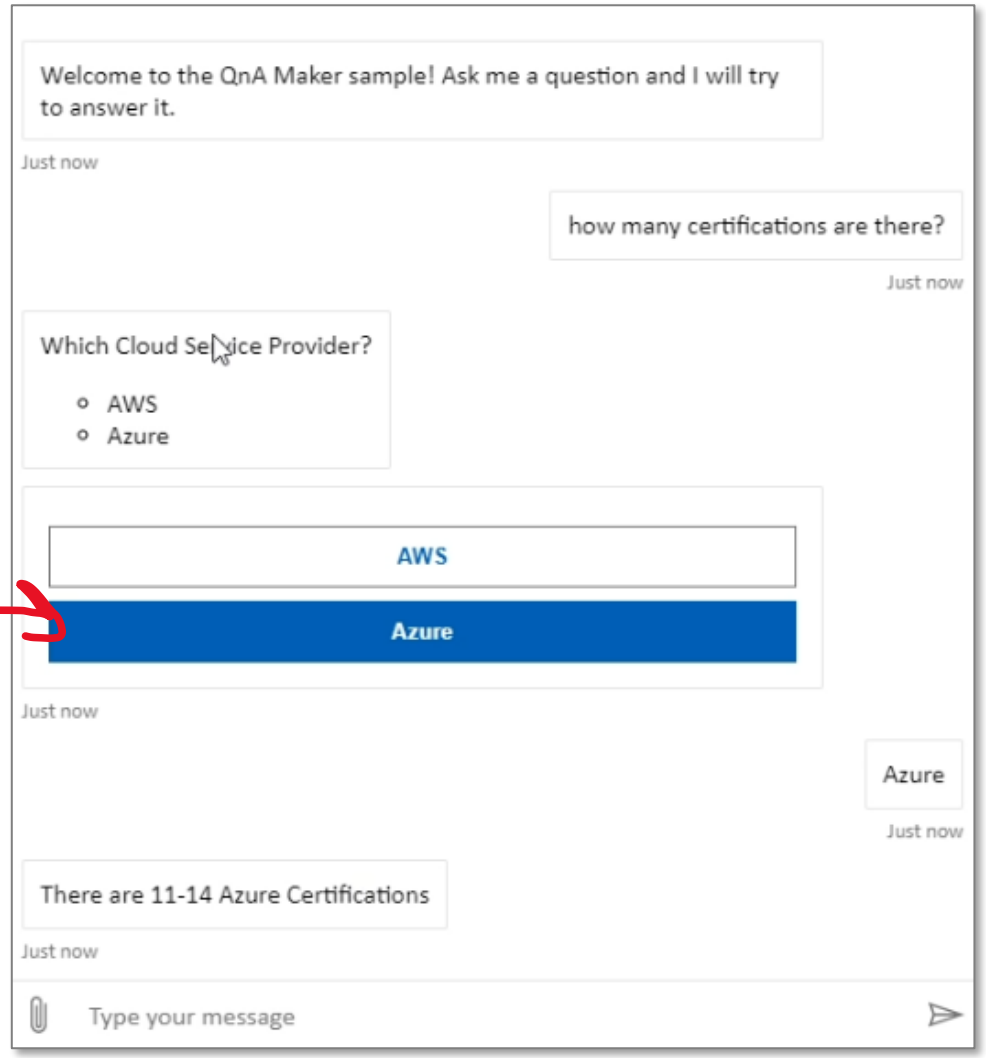


QnA Maker Service – Chat box

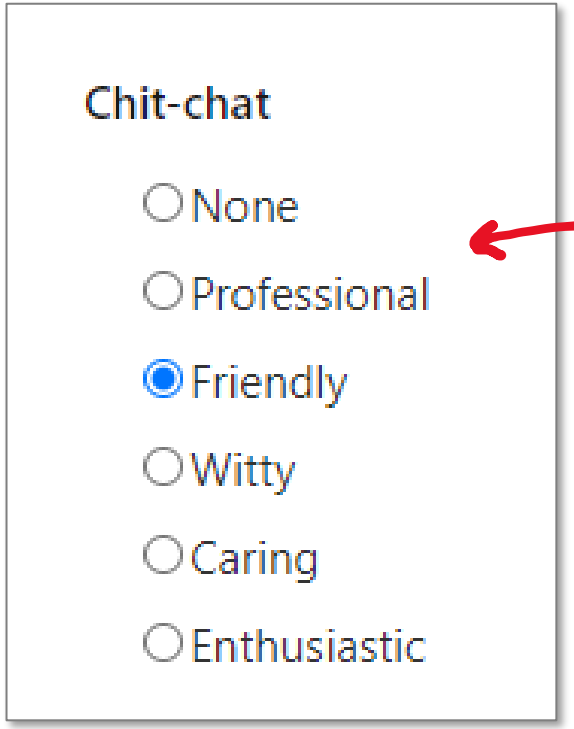
You converse with your bot through a Chat Box. There are many oppuritines to interact with your bot in QNAMaker.ai, Azure Bot Service, Bot Composer.

Via Channels you can even get embeddable chatbox code

Multi-turn conversation



QnA Maker Service – Chit Chat



The chit-chat feature in QnA maker allows you to easily add a **pre-populated set of the top chit-chat**, into your knowledge base.

This dataset has about **100 scenarios** of chit-chat in the voice of multiple personas

QnA Maker Service – Layered Ranking

QnA Maker's system is a layered ranking approach.

The data is stored in Azure search, which also serves as the first ranking layer.

The top results from Azure search are then passed through QnA Maker's NLP re-ranking model to produce the final results and confidence score.

QnA Maker Service – Multi-turn conversation

Multi-turn conversation is **follow-up prompts** and context to manage the multiple turns, known as *multi-turn*, for your bot from one question to another

When a question **can't be answered in a single turn**

QnA Maker provides multi-turn prompts and active learning to help you improve your basic question and answer pairs.

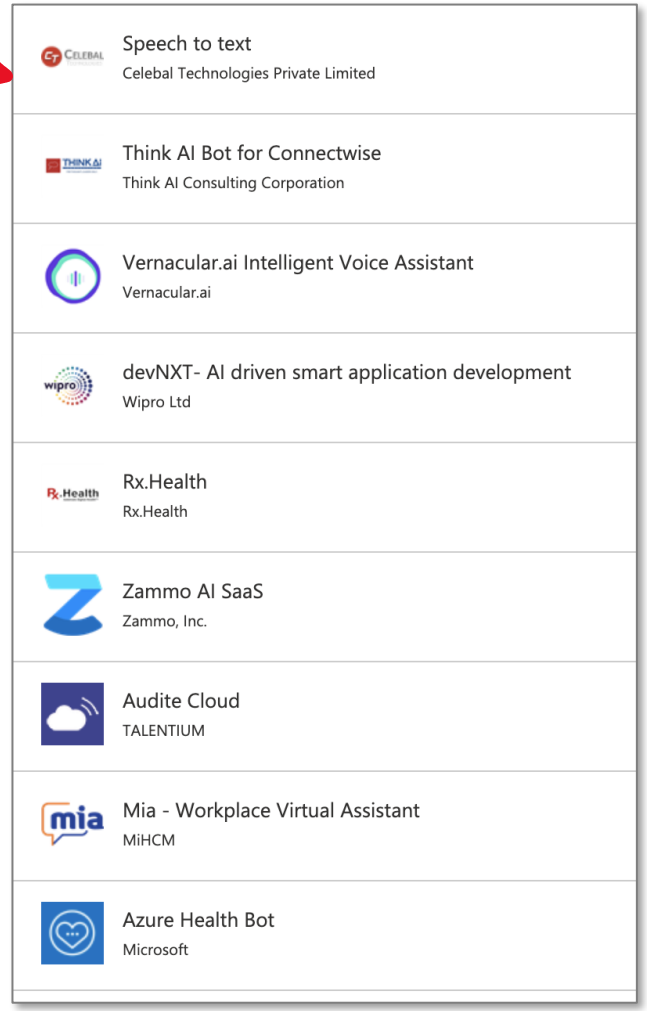
Multi-turn prompts give you the opportunity to connect question and answer pairs. This connection allows the client application to provide a top answer and provides more questions to refine the search for a final answer.

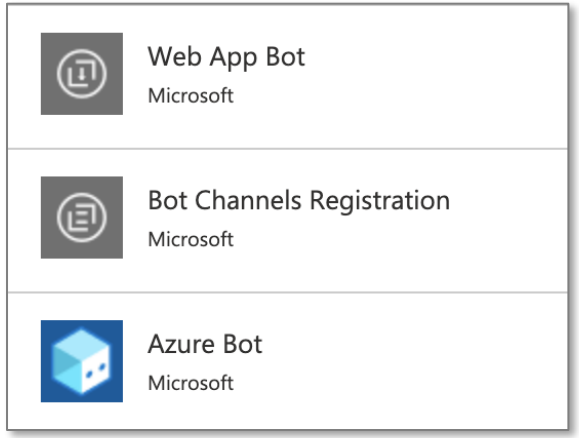
After the knowledge base receives questions from users at the published endpoint, QnA Maker applies **active learning** to these real-world questions to suggest changes to your knowledge base to improve the quality.

Azure Bot Service

Azure Bot Service Intelligent, serverless bot service that scales on demand.
Used for **creating, publishing, and managing bots**

You can **register and publish** a variety of bots from the Azure Portal





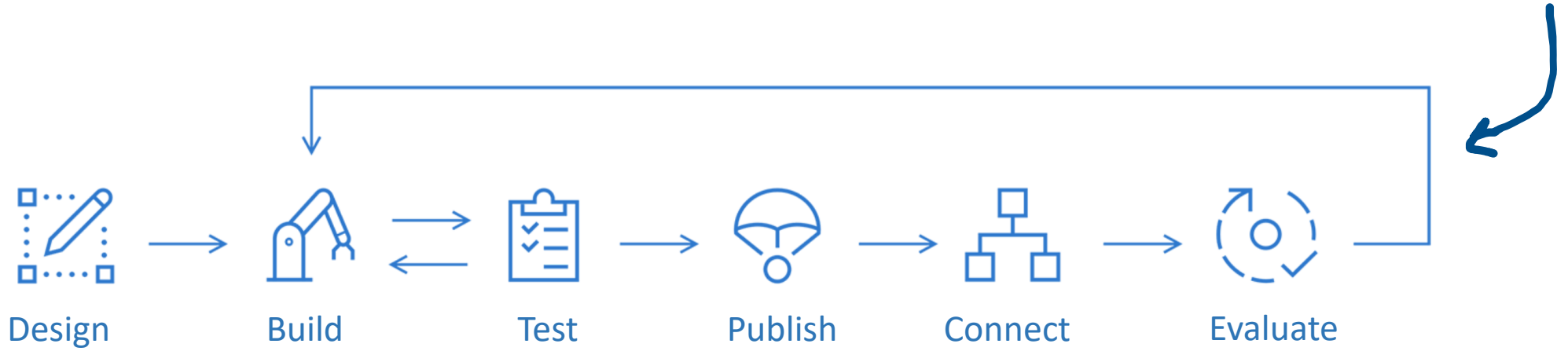
Azure Bot Service can integrate your bot with other Azure, Microsoft or Third Party services via **Channels:**

- Direct Line
- Alexa
- Office 365 email
- Facebook
- Kik
- LINE
- Microsoft Teams
- Skype
- Twilo
- ande more….

Bot Framework SDK

The Bot Framework SDK v4 is an **open-source SDK** that enable developers to **model and build sophisticated conversations**

The Bot Framework, along with the Azure Bot Service, provides and **end-to-end workflow**:



With this framework, developers can create bots that use speech, understand natural language, handle questions and answers, and more.

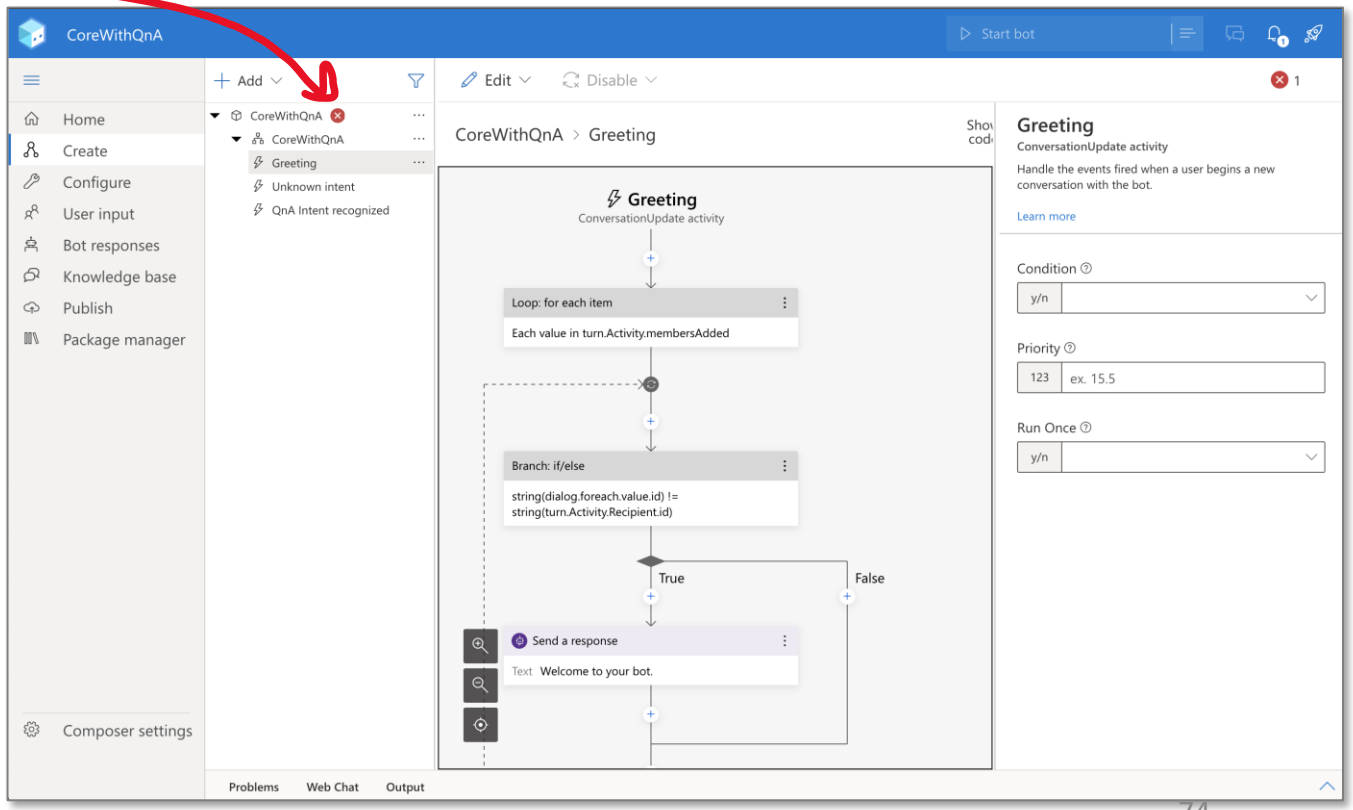
The Bot Framework includes a modular and extensible SDK for building bots, as well as tools, templates, and related AI services.

Bot Framework Composer

Bot Framework Composer, built on the Bot Framework SDK, is an **open-source IDE for developers** to **author, test, provision and manage** conversational experiences.

Composer is downable app available for Windows, OSX and Linux

- You can use either C# or Node to build your bot
- Deploy your bots to:
  - Azure Web App
  - Azure Functions
- Templates to build:
  - QnA Maker Bot
  - Enterprise or Personal Assistant Bot
  - Language Bot
  - Calendar or People Bot
- Test and debug via the Bot Framework Emulator
- Built in Package manager



Azure Machine Learning Service

Azure Machine Learning Studio (classic)
An older service that manages AI/ML workloads. Does not have a pipeline and other limitations. Workloads are not easily transferable to from classic to the new service.

Azure Machine Learning Service

A service that simplifies running AI/ML related workloads allowing you to build flexible Automated ML Pipelines. Use Python or R, Run DL workloads such as Tensorflow

Jupyter Notebooks
- build and document your machine learning models as you build them, share and collaborate

Azure Machine Learning SDK for Python
- An SDK designed specifically to interact with Azure Machine Learning Services

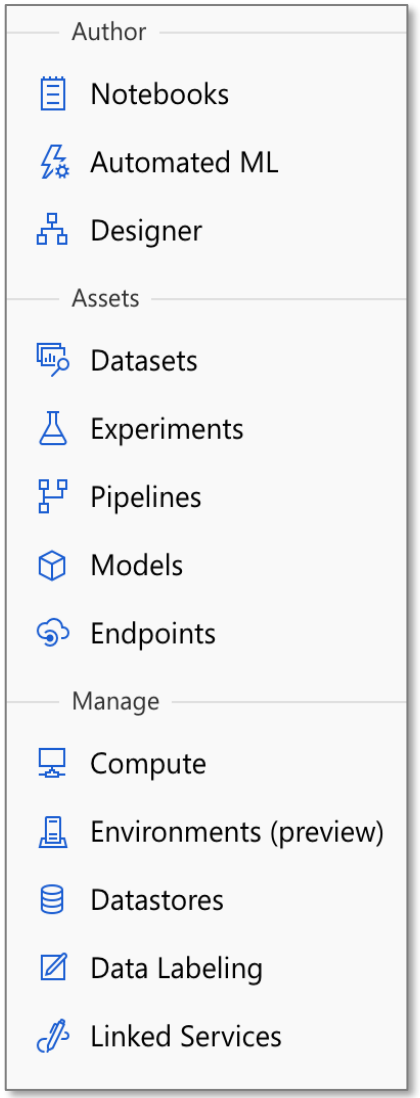
MLOps
- end to end automation of ML model pipelines eg. CI/CD, training, inference

Azure Machine Learning Designer
- drag and drop interface to visually build, test, and deploy machine learning models

Data Labeling Service
- ensemble a team of humans to label your training data

Responsible Machine Learning
- model fairness through disparity metrics and mitigate unfairness

Azure Machine Learning Studio – Overview



Author

Notebooks — Jupyter Notebooks, an IDE to write python code to build ML models
AutoML — Completely automated process to build and train an ML model
Designer — Visual drag and drop designer to construct end to end ML pipelines

Assets

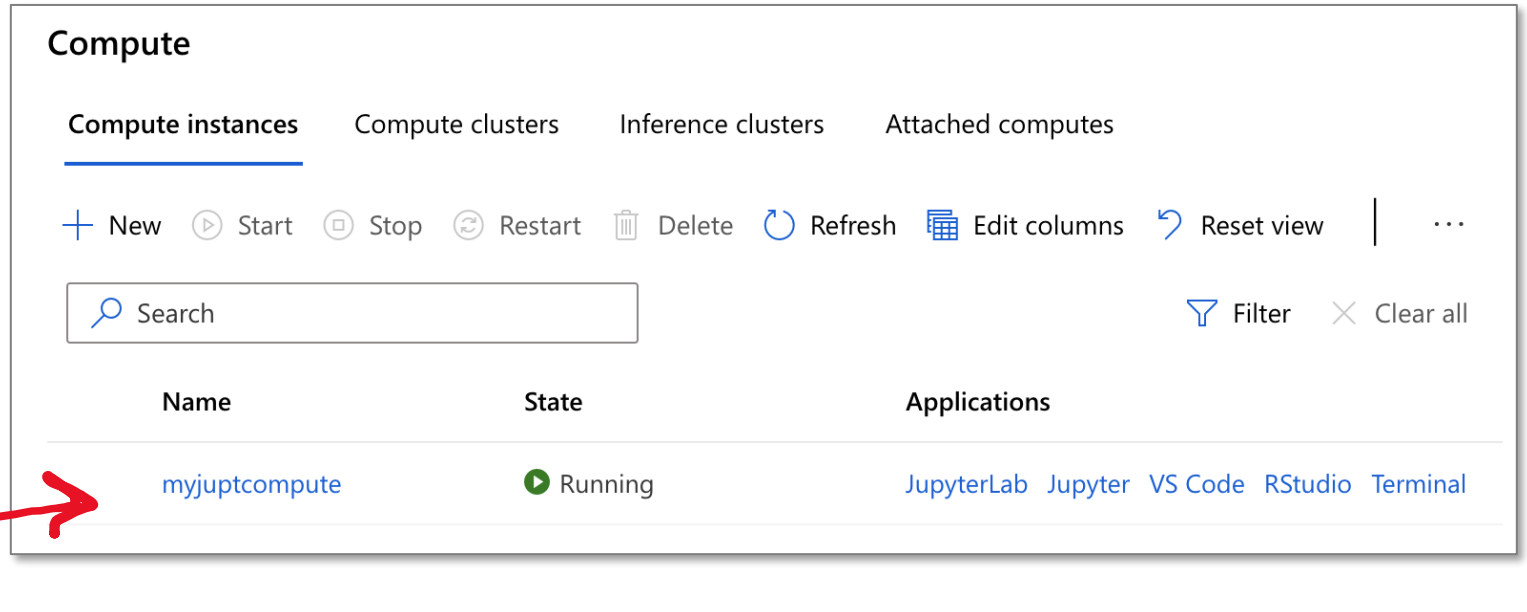
Datasets — data that you upload which will be used for training
Experiments — when you run a training job they are detailed here
Pipelines — ML workflows you have built, or you have used in the Designer
Models — a model registry containing trained models that can be deployed
Endpoints — when you deploy a model its hosted on an accessible endpoint eg. REST API

Manage

Compute — the underlying computing instances used to for notebooks, training, inference
Environments — a reproducible Python environment for machine learning experiments
Datastores — a data repository where your dataset resides
Data Labeling — have humans with ML-assisted labeling to label your data for supervised learning
Linked Services — external services you can connect to the workspace eg. Azure Synapse Analytics

Azure Machine Learning Studio – Compute

Azure Machine Learning Studio has **4 kinds of compute**:



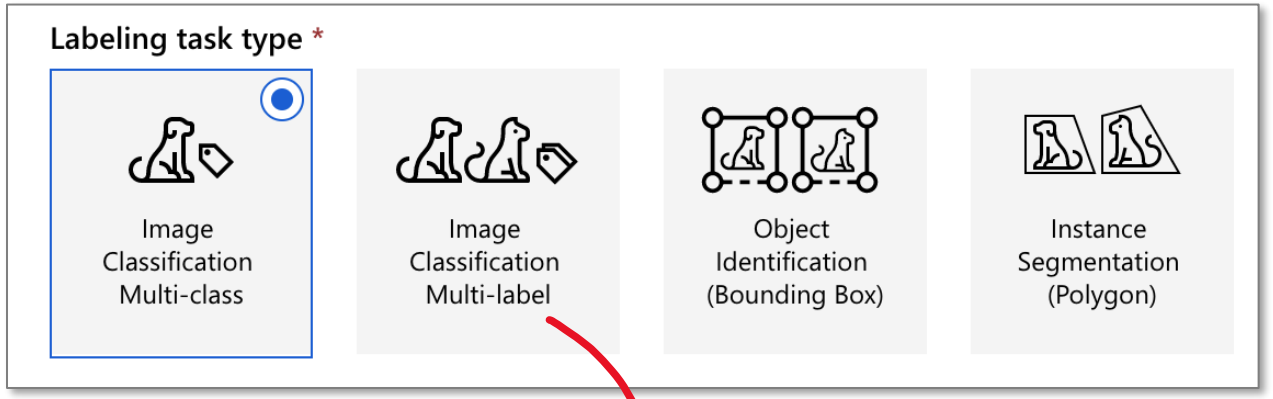
1. **Compute Instances** — Development workstations that data scientists can use to work with data and models.
2. **Compute Clusters** — Scalable clusters of virtual machines for on-demand processing of experiment code.
3. **Inference Clusters** — Deployment targets for predictive services that use your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

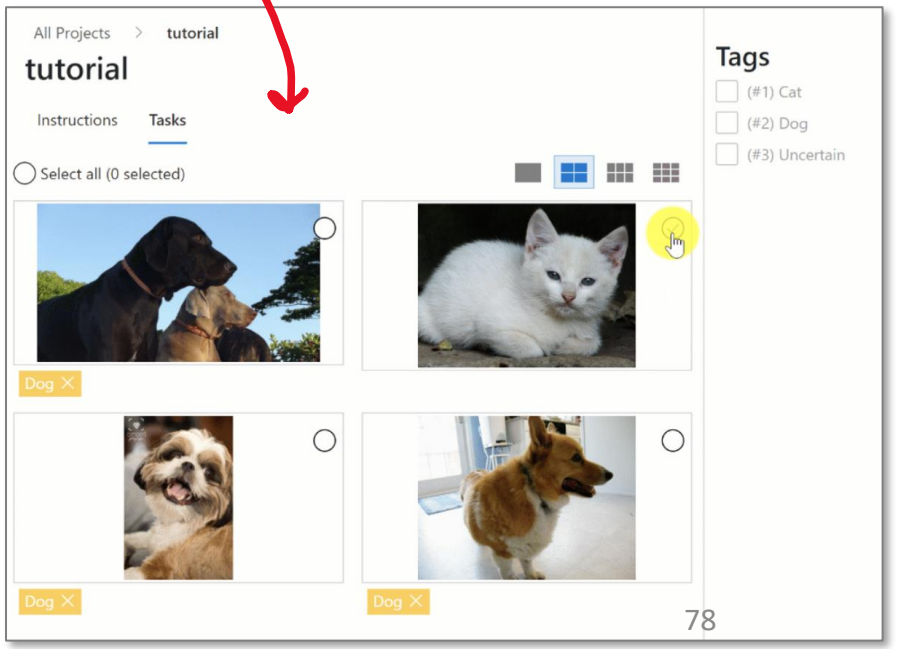
Azure Machine Learning Studio – Data Labeling

Create Data labeling jobs to prepare your Ground Truth for supervised learning

Human-in-the-loop labeling
You have a team of humans that will apply labeling
These are humans you grant access to labeling

Machine-learning-assisted data labeling
You will use ML to perform labeling



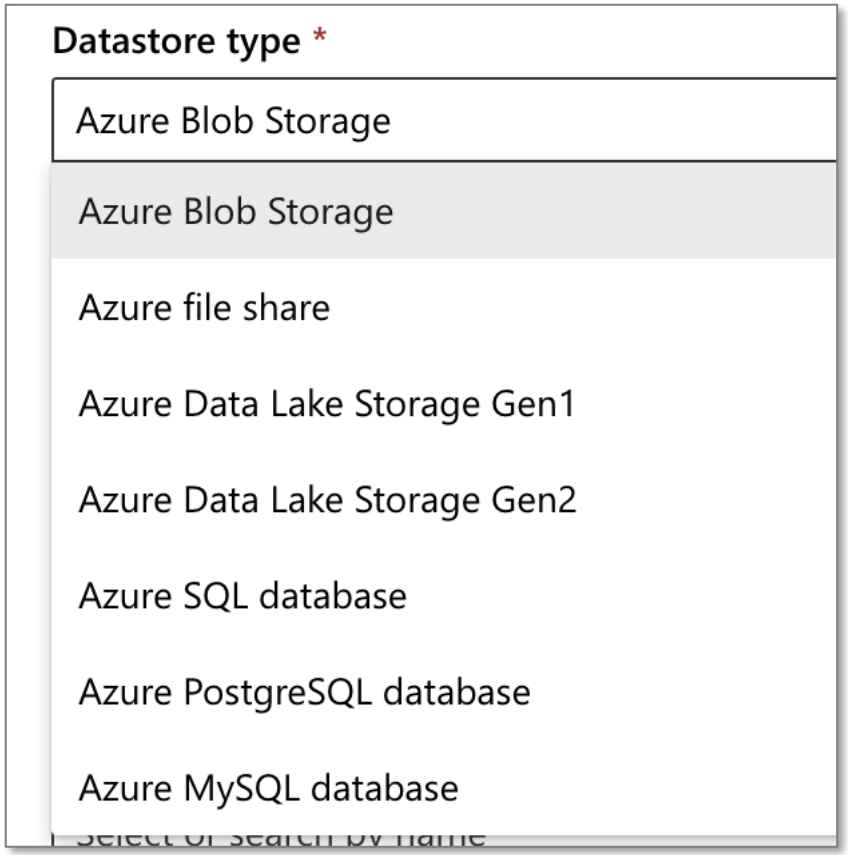


You can export the label data for Machine Learning experimentation at any time
Users often export multiple times and train different models, rather than wait for all the images to be labeled.
Image labels can be exported in:

- COCO format
- Azure Machine Learning dataset
  - dataset format makes it easy to use for training in Azure Machine Learning

Azure Machine Learning Studio – Data Stores

Datastores securely connect to your storage service on Azure without **putting your authentication credentials** and the integrity of your original data source **at risk**.



Azure Blob Storage
data is stored as objects, distributed across many machines

Azure File Share
a mountable file share via SMB and NFS protocols

Azure Data Lake Storage (Gen 2)
Azure Blob storage designed for vasts amount of data for Big Data analytics

Azure SQL database
Full-managed MS SQL relational database

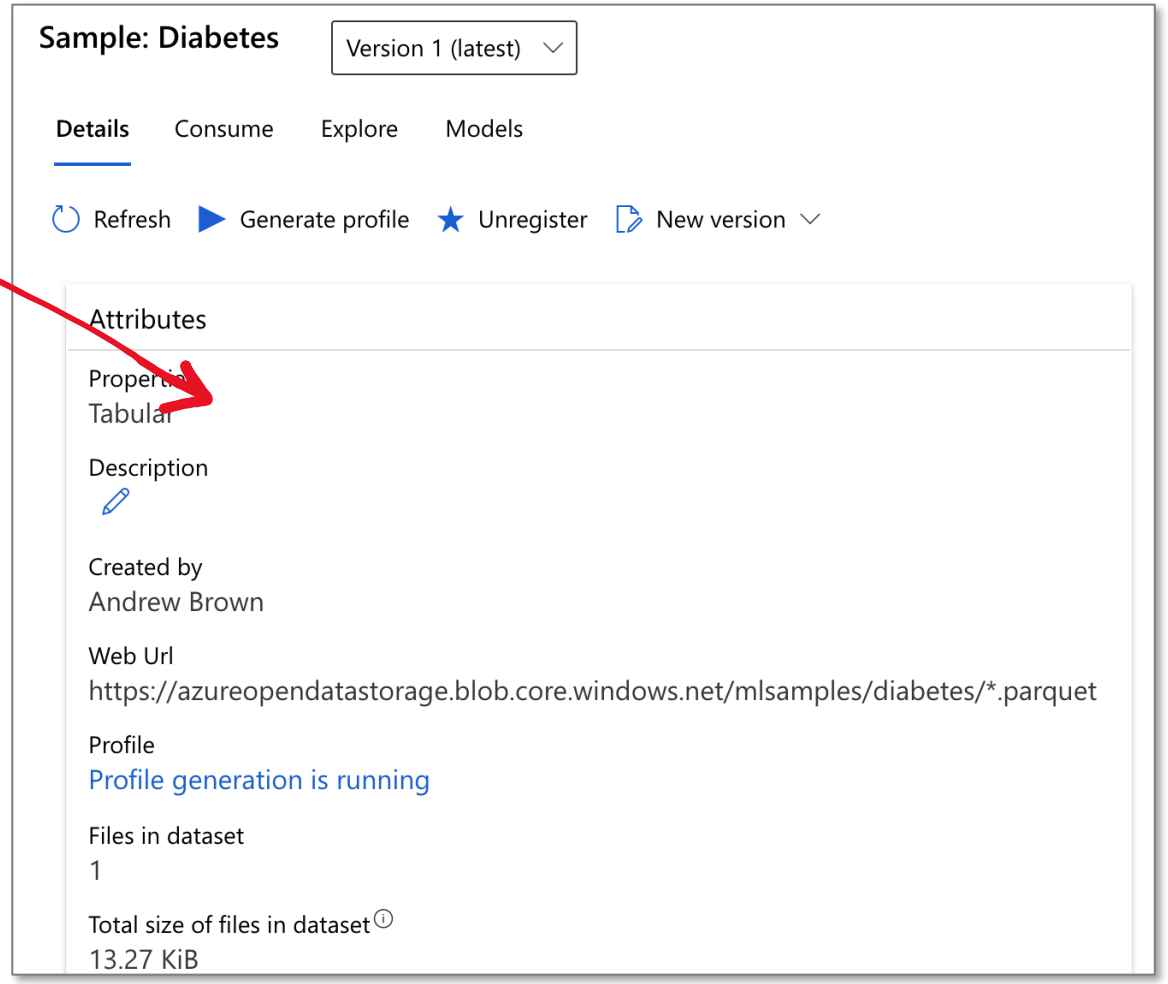
Azure Postgres database
open-source relational database

Azure MySQL Database
Open-source relational database

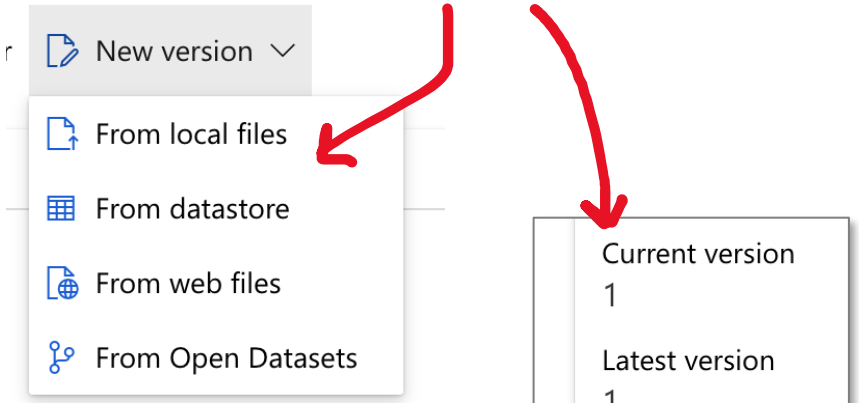
Azure Machine Learning Studio – Datasets

Azure ML Datasets makes it **easy to register your datasets** for use with your ML workloads

There will be various metdata associated to your dataset



You can upload new datasets and they will be **versioned**



Azure Machine Learning Studio – Datasets

Azure provides a same code snippet with the **Azure Machine Learning SDK for Python** to **start programmatically using datasets in your Jupyter Notebooks**

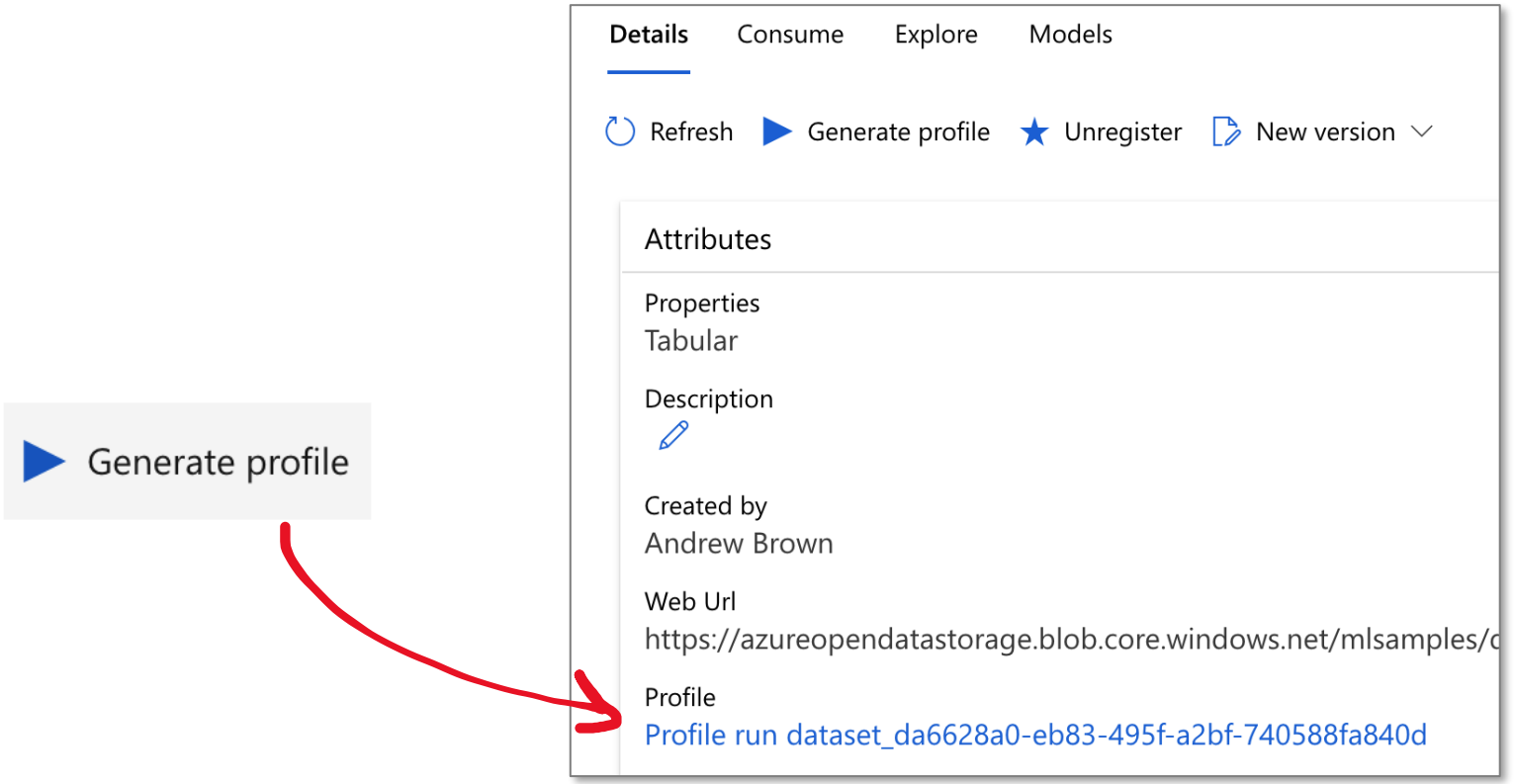
Sample usage

```
# azureml-core of version 1.0.72 or higher is required
# azureml-dataprep[pandas] of version 1.1.34 or higher is required
from azureml.core import Workspace, Dataset

subscription_id = '7f3352cf-6c7d-456a-8ecb-83ef2128907b'
resource_group = 'MyStudio'
workspace_name = 'MyStudio'

workspace = Workspace(subscription_id, resource_group, workspace_name)

dataset = Dataset.get_by_name(workspace, name='Sample: Diabetes')
dataset.to_pandas_dataframe()
```

Azure Machine Learning Studio – Datasets

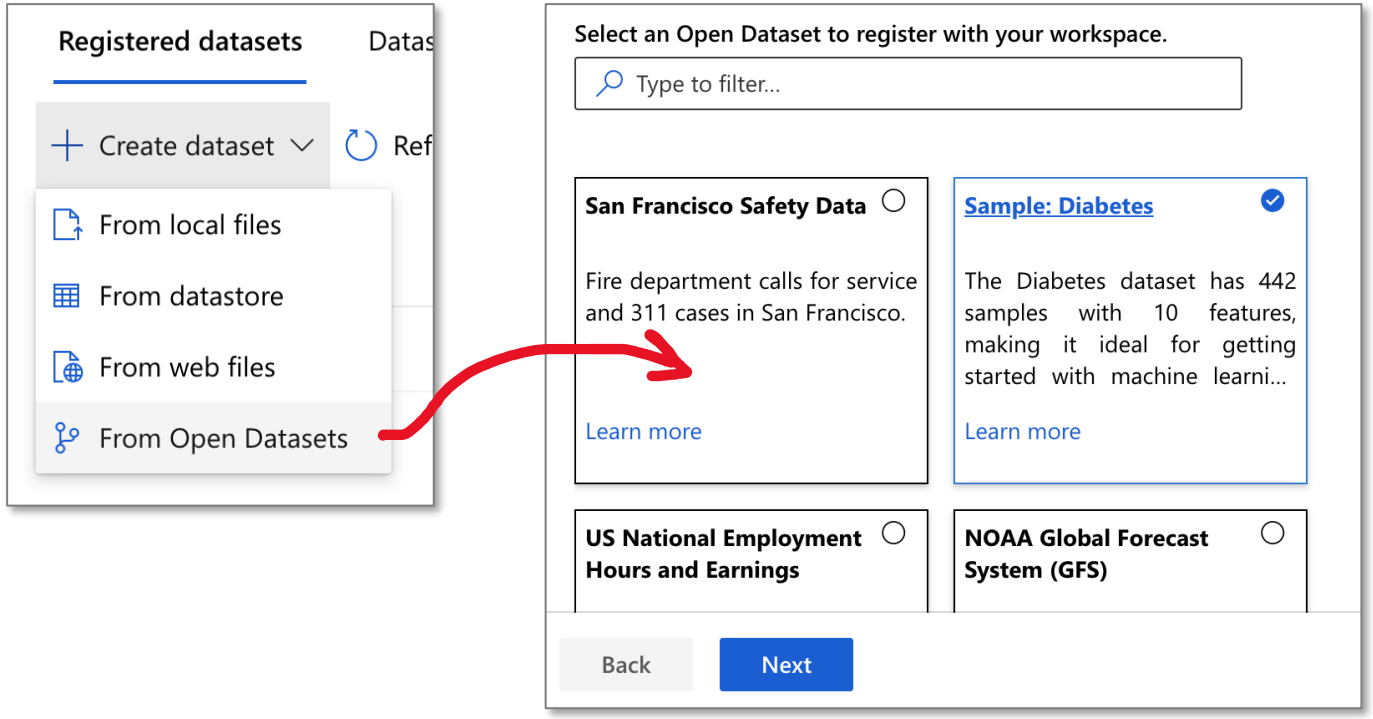
Generate Profile

You can create a data profile that has summary statistics, distribution of the data and more.
You will need to launch a compute instance to generate a profile.



Azure Machine Learning Studio – Open Datasets

Open DataSets are **publicly hosted datasets** that are commonly **used for learning how to build ML models**

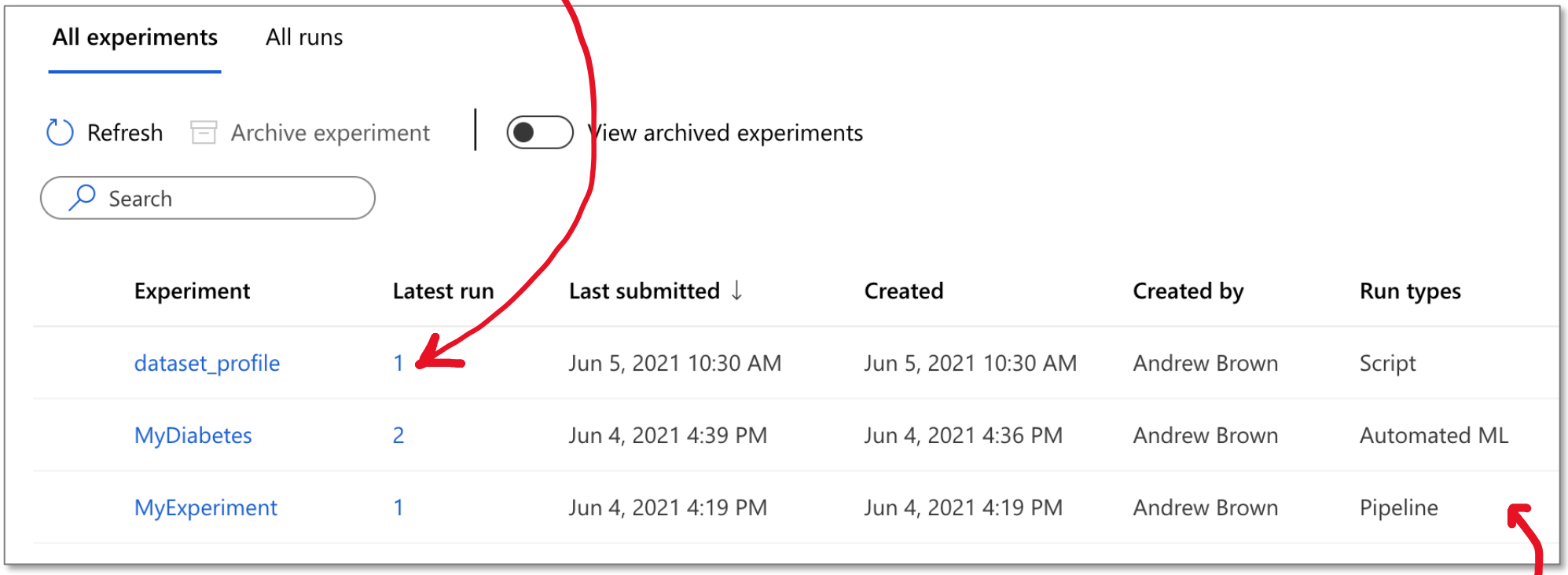


Azure has a curated list of open-datasets that you can quickly add to your data store.
Great for learning how to use AutoML or Azure Machine Learning Designer

Azure Machine Learning Studio – Experiments

Experiments is a **logical grouping Azure Runs**

Runs are the **act of running an ML task** on a virtual machine or container



Experiments do not include *Inference*

The contents of a run will vary based o its **Run Type**

Azure Machine Learning Studio – Pipelines

Azure ML Pipelines is an **executable workflow** of a complete machine learning task

Subtasks are encapsulated as a series of steps within the pipeline

Independent steps allow multiple data scientists to work on the same pipeline at the same time without over-taxing compute resources

Separate steps also make it easy to use different compute types/sizes for each step.

When you rerun a pipeline, the run jumps to the steps that need to be rerun, such as an updated training script.

Steps that do not need to be rerun are skipped

After a pipeline has been published, you can configure a REST endpoint, which allows you to rerun the pipeline from any platform or stack

You can build pipelines two ways:

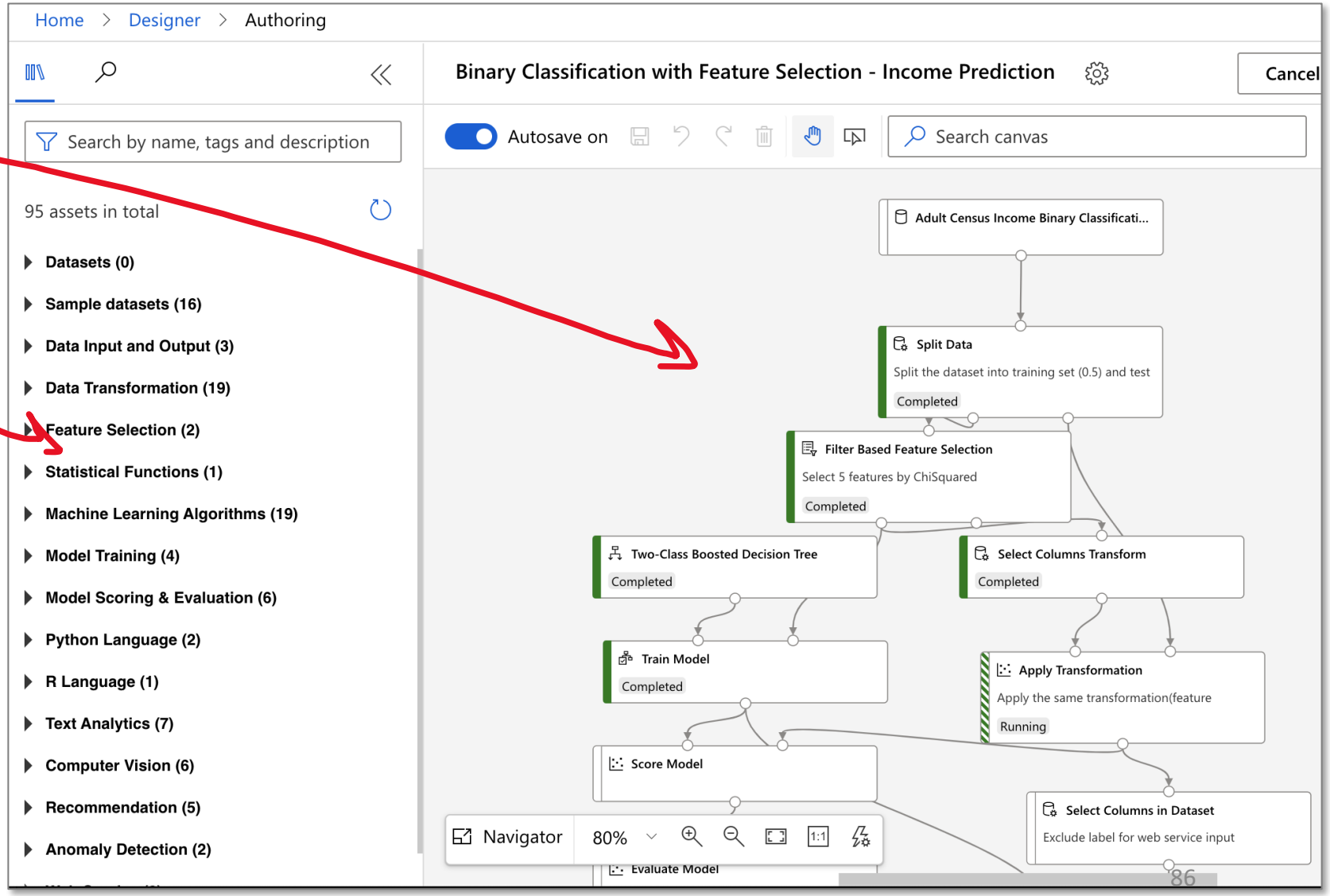
- Using the Azure Machine Learning Designer
- **Programmatically using the Azure Machine Learning Python SDK**

```
ws = Workspace.from_config()
blob_store = Datastore(ws, "workspaceblobstore")
compute_target = ws.compute_targets["STANDARD_NC6"]
experiment = Experiment(ws, 'MyExperiment')

input_data = Dataset.File.from_files(
    DataPath(datastore, '20newsgroups/20news.pkl'))
prepped_data_path = OutputFileDatasetConfig(name="output_path")

dataprep_step = PythonScriptStep(
    name="prep_data",
    script_name="dataprep.py",
    source_directory="prep_src",
    compute_target=compute_target,
    arguments=["--prepped_data_path", prepped_data_path],
    inputs=[input_dataset.as_named_input('raw_data').as_mount() ]
    )

prepped_data = prepped_data_path.read_delimited_files()

train_step = PythonScriptStep(
    name="train",
    script_name="train.py",
    compute_target=compute_target,
    arguments=["--prepped_data", prepped_data],
    source_directory="train_src"
)
steps = [ dataprep_step, train_step ]

pipeline = Pipeline(workspace=ws, steps=steps)

pipeline_run = experiment.submit(pipeline)
pipeline_run.wait_for_completion()
```

Azure Machine Learning Studio – ML Designer

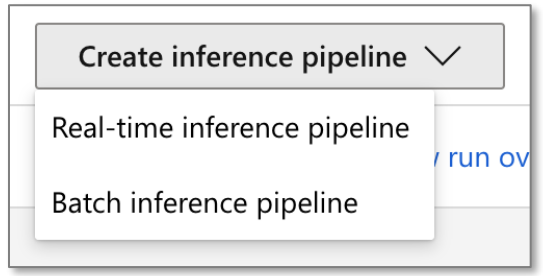
The Azure Machine Learning Designer lets you **quickly build Azure ML Pipelines without having to write code.**

You can **drag out various templated steps** called assets to quickly prototype your pipeline

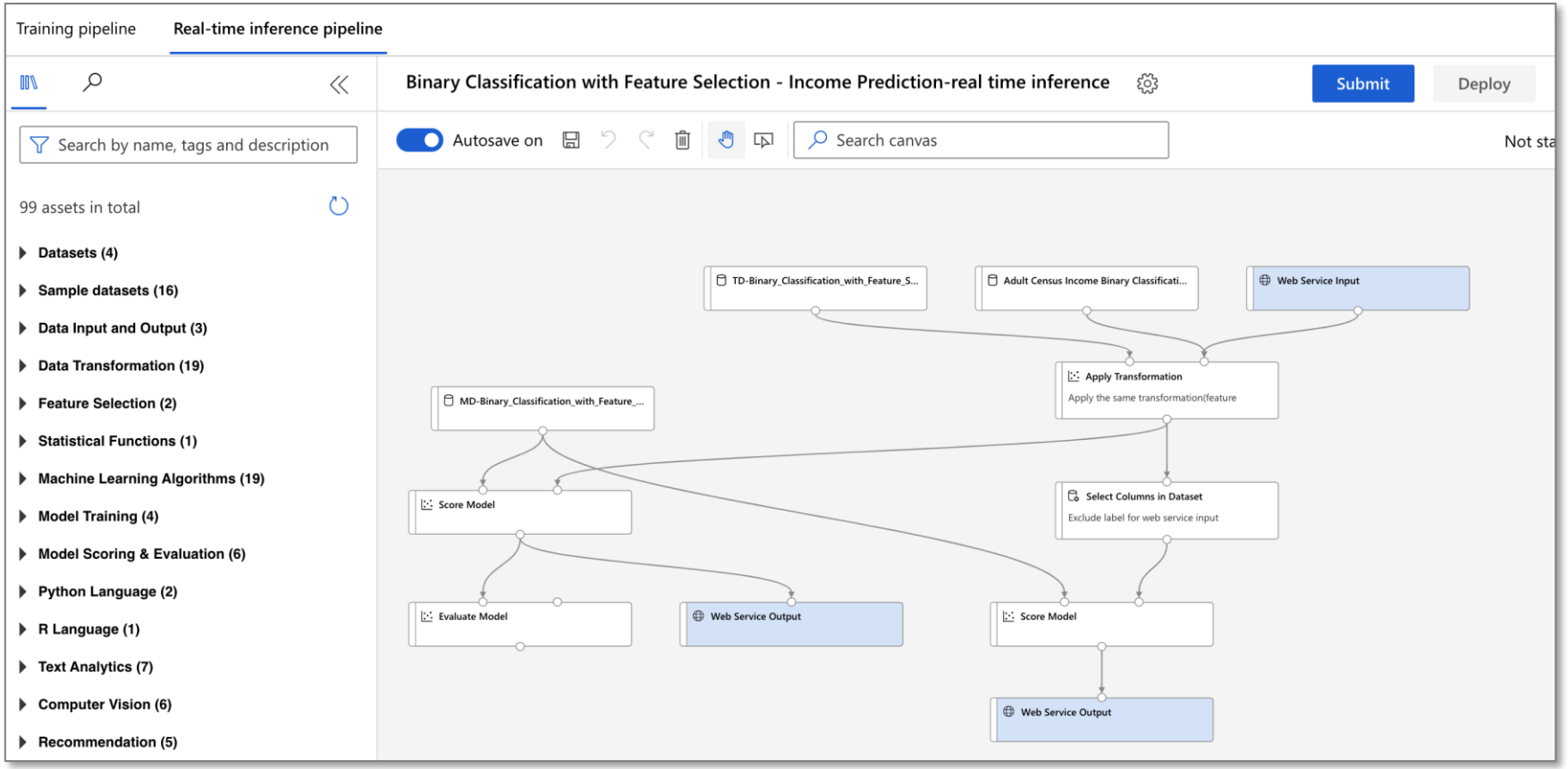


Azure Machine Learning Studio – Machine Learning Designer

One a pipeline is trained you can create an **inference pipeline**



You can **toggle** between your training in inference pipeline

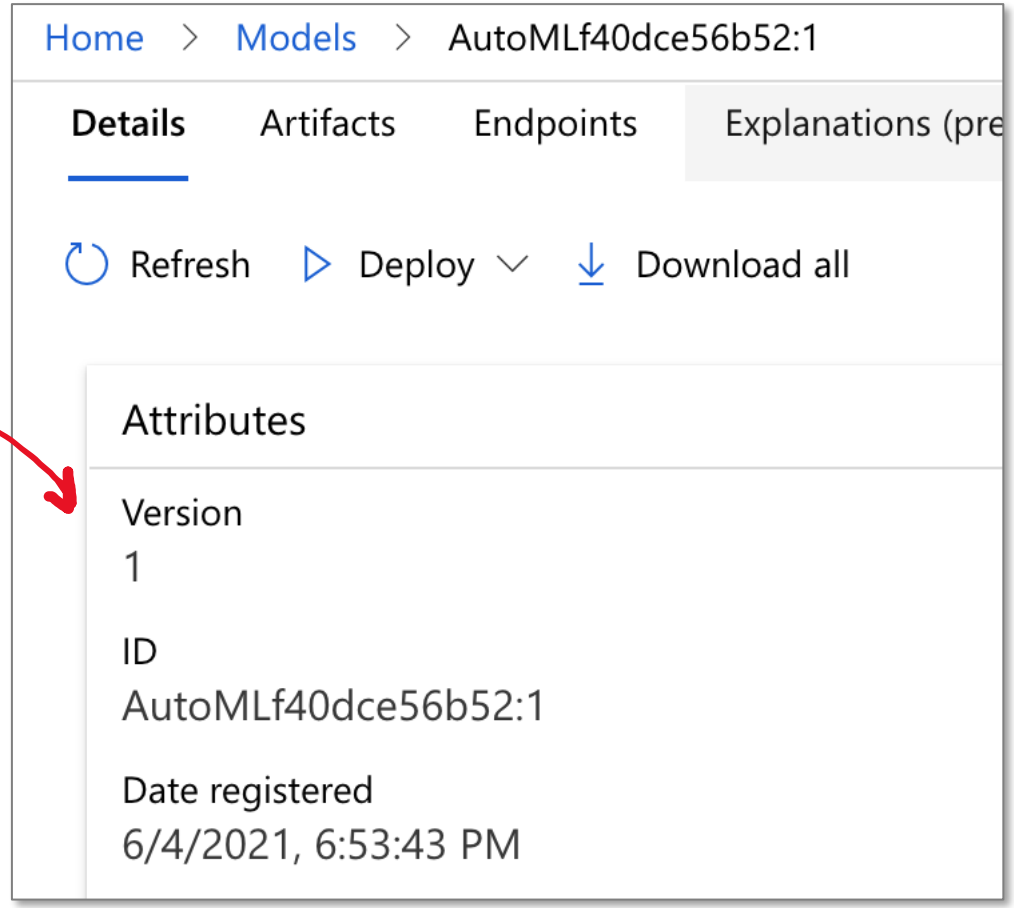


Azure Machine Learning Studio – Models

Model Registry allows you to **create, manage and track your registered models** as incremental versions under the same name

Each time you register a model with the same name as an existing one, the registry assures that **it's a new version.**

Additionally, you can provide metadata tags and use the tags when you search for models.



Azure Machine Learning Studio – Endpoints

Azure ML Endpoints allow you to **deploy machine learning models as a web service**

The workflow for deploying a model:

- Register the model
- Prepare an entry script
- Prepare an inference configuration
- Deploy the model locally to ensure everything works
- Choose a compute target
- Re-deploy the model to the cloud
- Test the resulting web service

Realtime endpoints

An endpoint that provides remote access to invoke the ML model service running on either:

- Azure Kubernetes Service (AKS)
- Azure Container Instance (ACI)

Pipeline endpoints

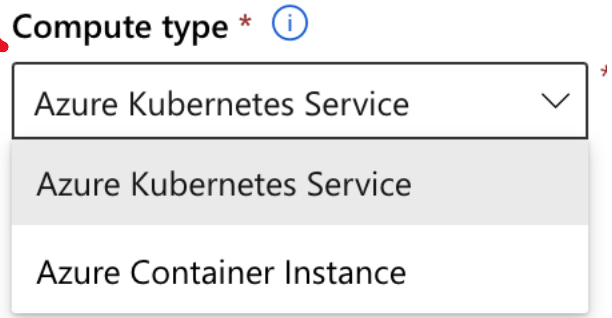
An endpoint that provide remote access to invoke an ML pipeline.

You can parametrize the pipeline endpoint for managed repeatability in batch scoring and retraining scenarios.

Azure Machine Learning Studio – Endpoints

When you **deploy a model** to an endpoint it will either be deployed to:

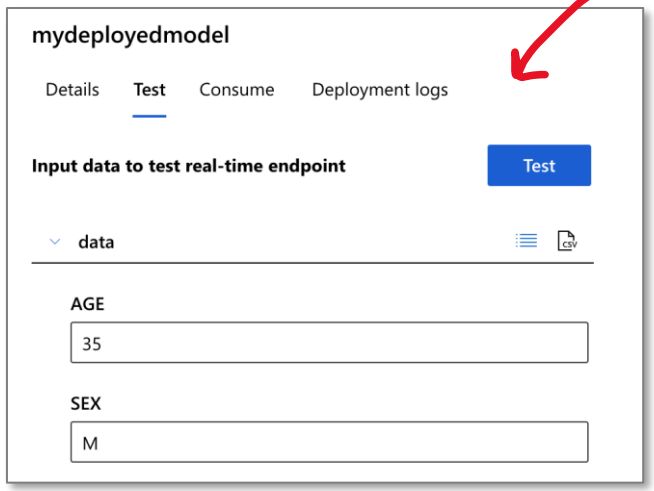
- Azure Kubernetes Service (AKS)
- Azure Container Instance (ACI)

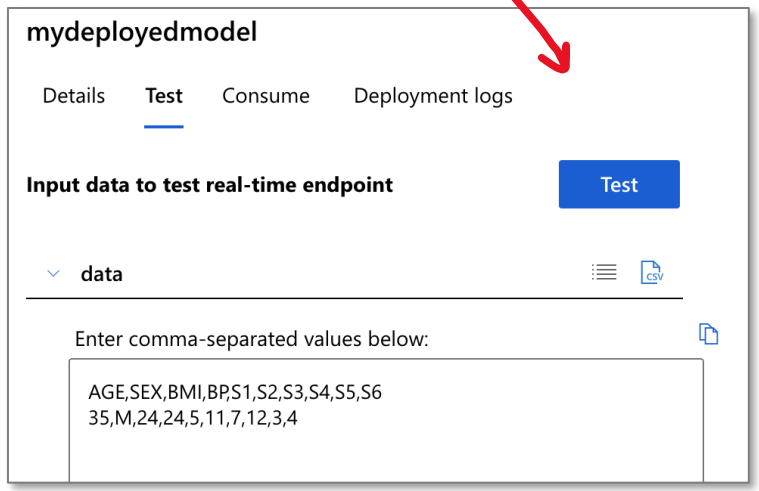


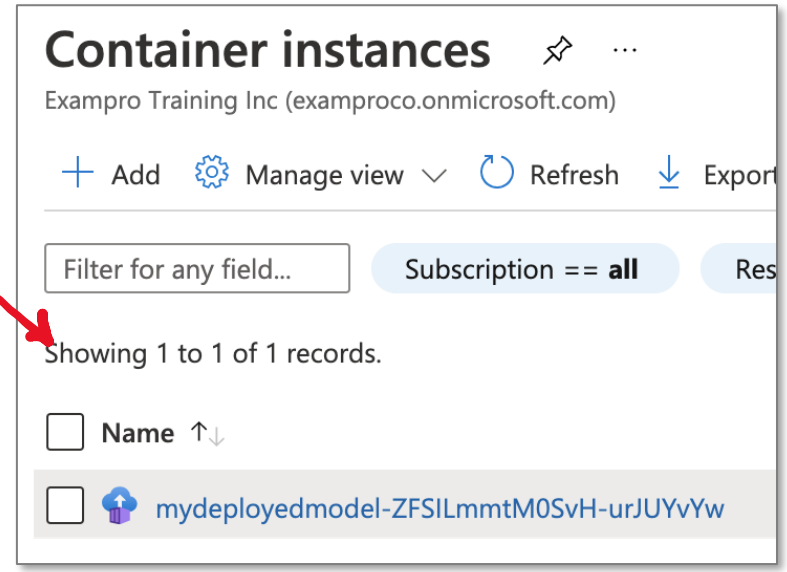
The computing resource will not show in Azure Machine Learning Studio

You need to check AKS or ACI

When you have deployed a real-time endpoint you can test the endpoint by sending a **single request** or a **batch request**.

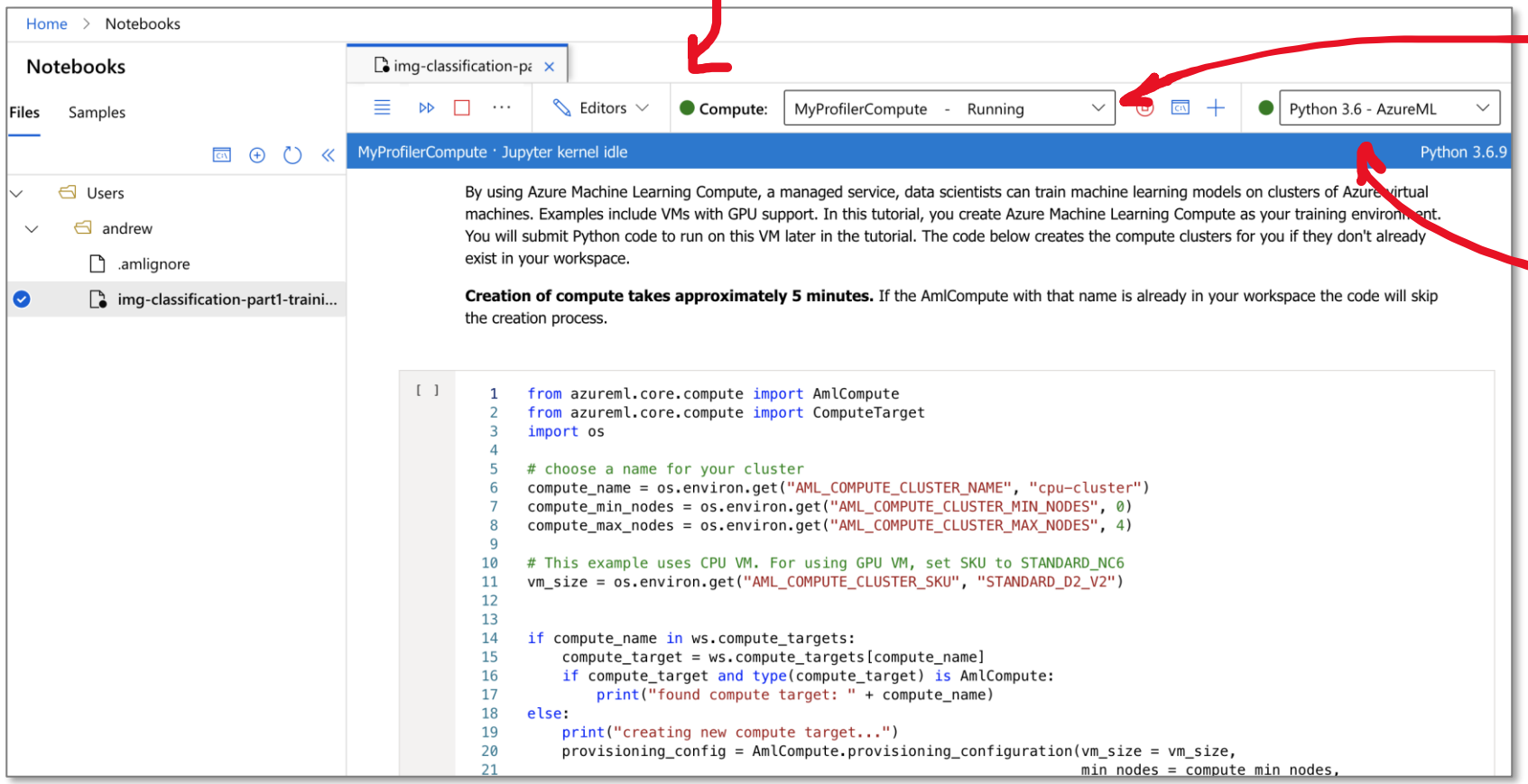






Azure Machine Learning Studio – Notebooks

Azure has a built in **Jupyter-like Notebook editor** so you can build and train your ML models



Choose Compute

You need to create a compute instance to run your Notebook

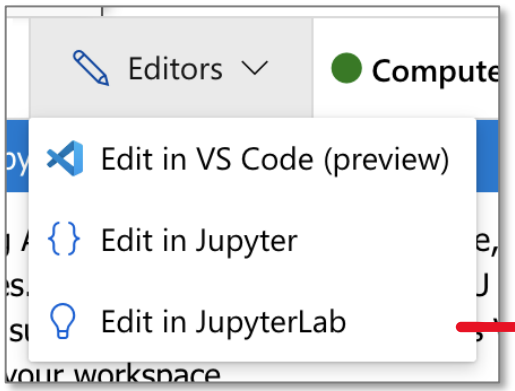
Choose Kernel

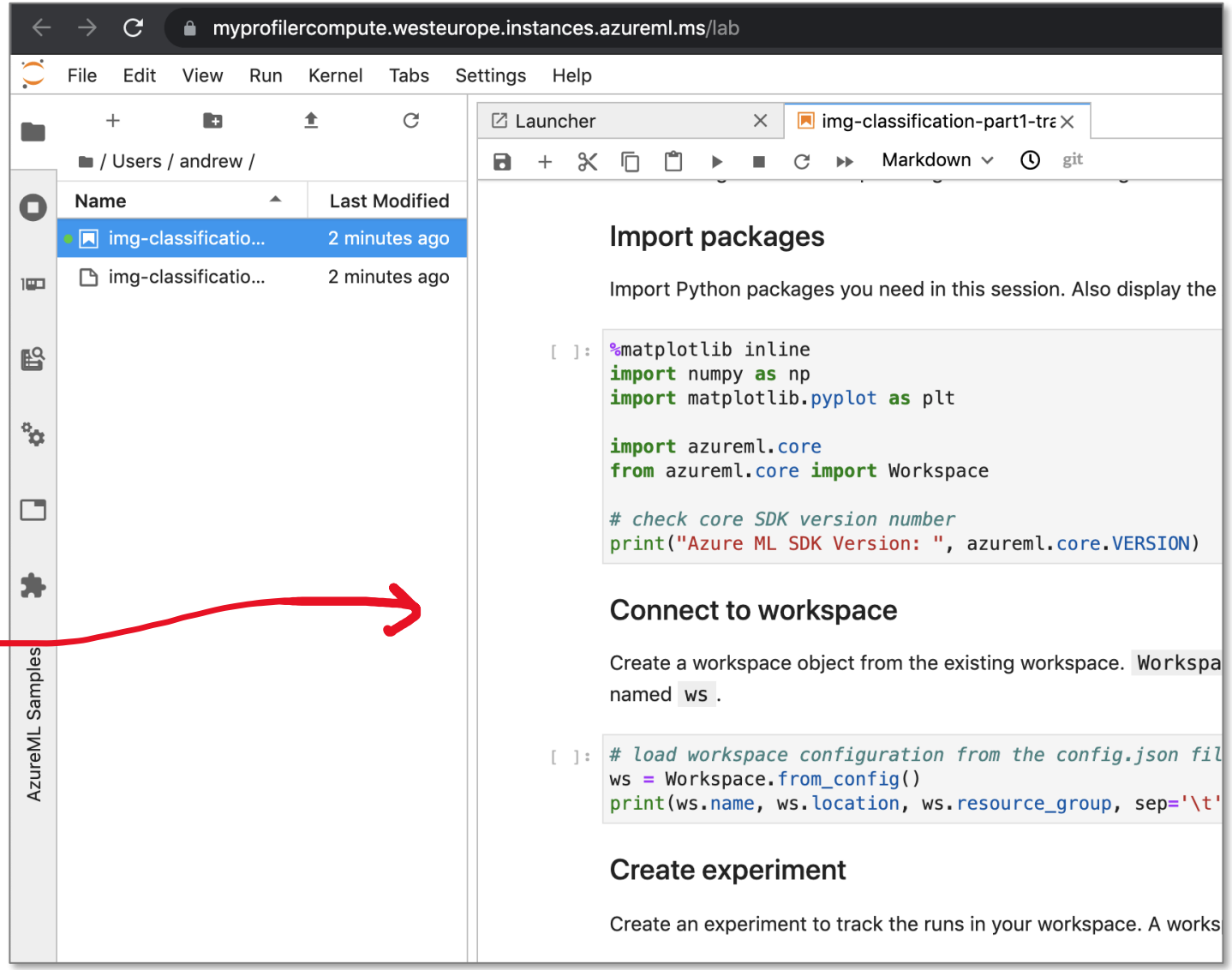
You need to choose a Kernel which preload a programming language and programming libraries for different use cases

Azure Machine Learning Studio – Notebooks

You can open the Notebook in a more familiar IDE:

- VSCode
- Jupyter Notebook (classic)
- **Jupyter Labs**





AutoML

Automated machine learning (AutoML) **automates the process of creating an ML model**.

With Azure AutoML you

- supply a dataset
- **Choose a Task Type** (Classification, Regression or Time Series Forecasting)
- Then AutoML will train and tune your model

Classification

When you need to make a prediction based on several classes:

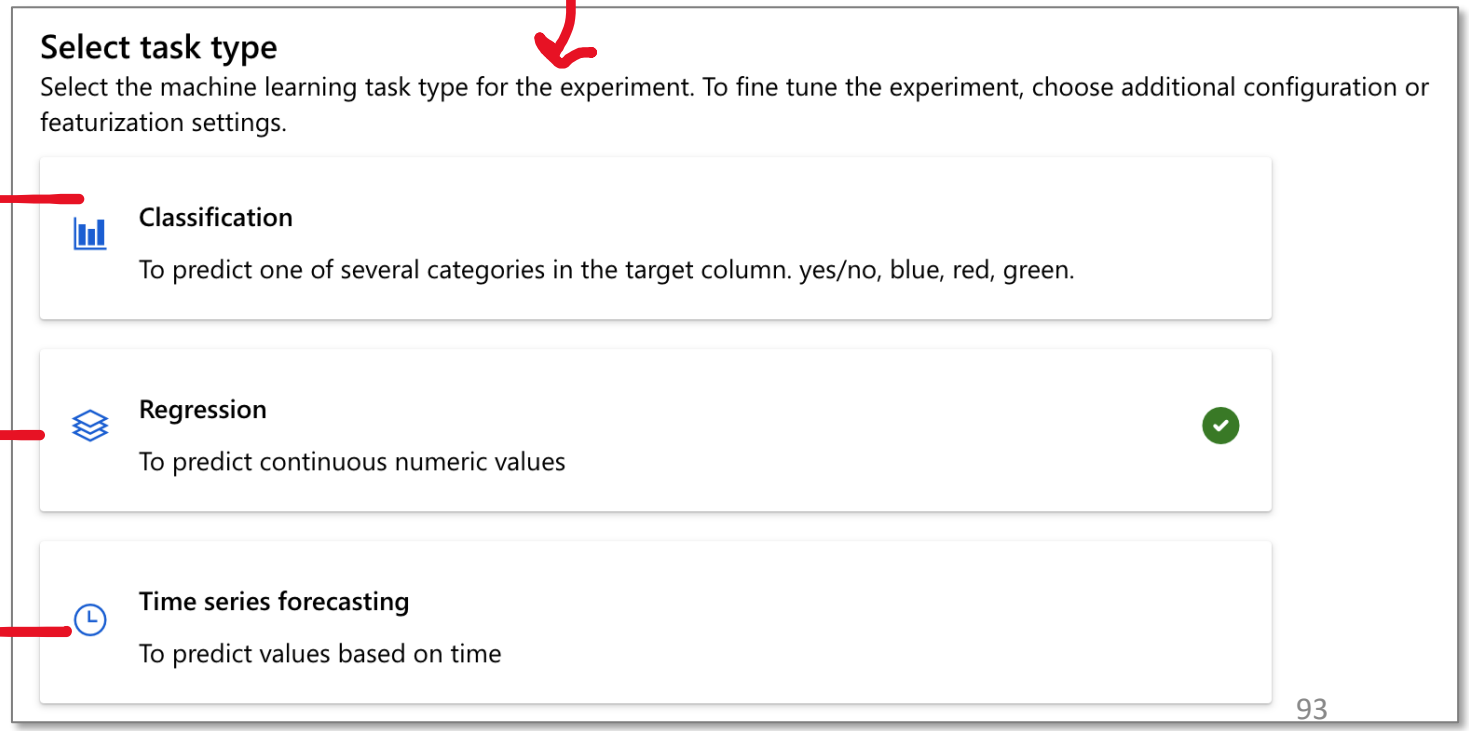
- binary classification: Yes or No
- multi-class classification: Red, Green, Blue

Regression

When you need to predict a continuous number value

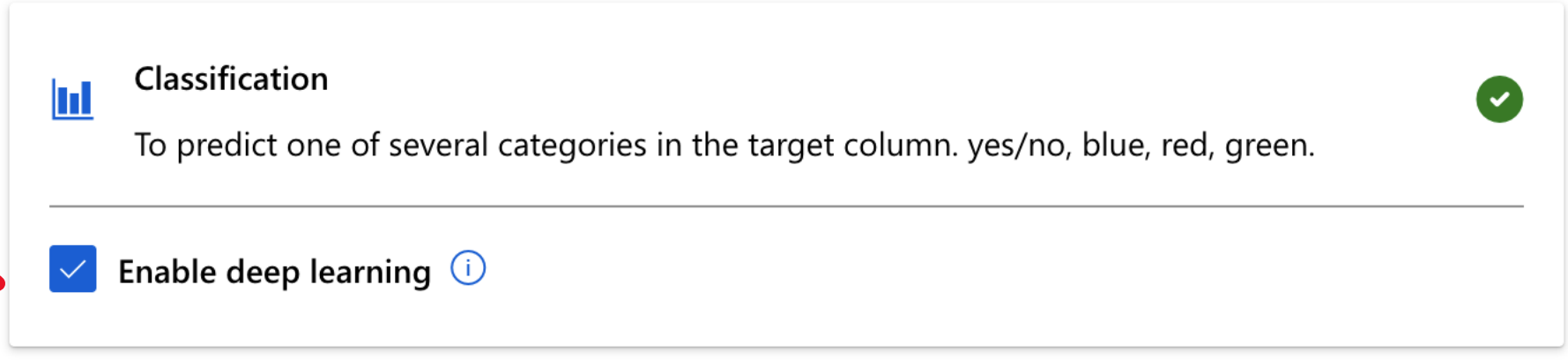
Time Series Forecasting

When you need to predict the value based on time



AutoML – Classification

Classification is a type of **supervised learning** in which **models learn using training data**, and apply those learnings to new data.



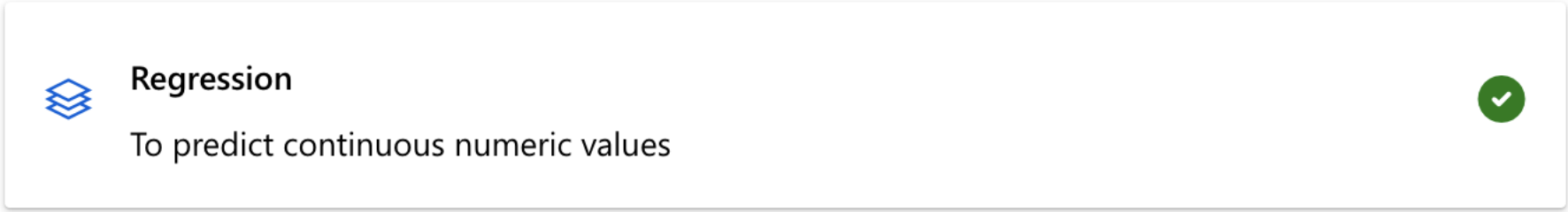
If you enable Deep Learning than you will likely want a **GPU compute**

The goal of classification models is to **predict which categories new data will fall into** based on learnings from its training data:

- **binary classification:** a record is labeled out of two possible labels eg: true or alse
- **multiclass classification**: a record is labeled out of range of labels: happy, sad, mad, rad

AutoML – Regression

Regression is a type of **supervised learning** in which **models learn using training data**, and apply those learnings to new data.



The goal of regression is to predict a variable in the future

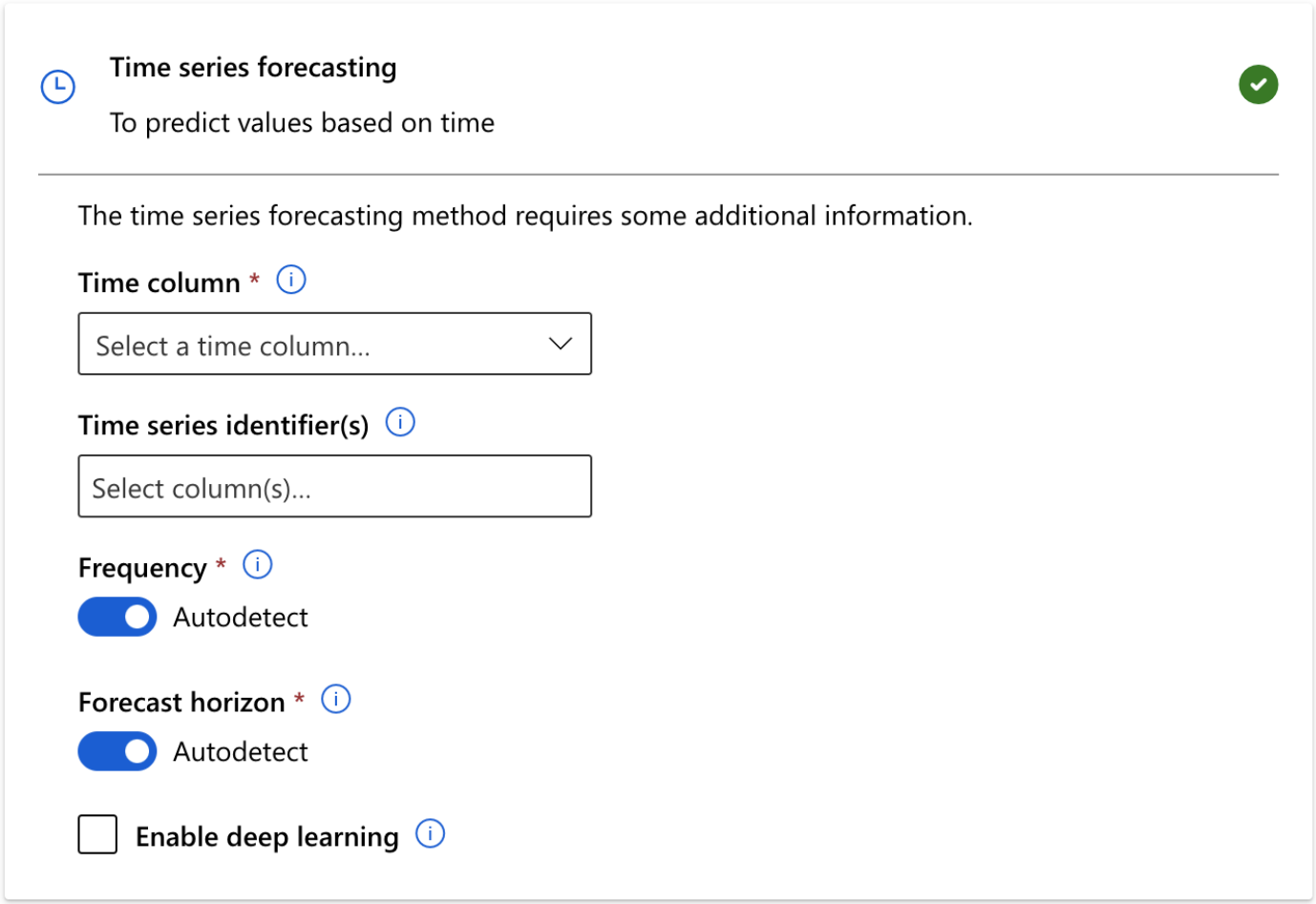
AutoML – Time Series Forecasting

Forecast revenue, inventory, sales, or customer demand

An automated time-series experiment is treated as a **multivariate regression problem**

Past time-series values are "pivoted" to become additional dimensions for the regressor together with other predictor

unlike classical time series methods, has an advantage of naturally incorporating multiple contextual variables and their relationship to one another during training



AutoML – Time Series Forecasting

Advanced forecasting configuration includes:

- holiday detection and featurization
- time-series and DNN learners (Auto-ARIMA, Prophet, ForecastTCN)
- many models support through grouping
- rolling-origin cross validation
- configurable lags
- rolling window aggregate features

AutoML – Data Guard Rails

Data guardrails are run by Azure AutoML when **automatic featurization** is enabled.
A **sequence of checks** to **ensure high quality input data** is being used to train model.

Type	Status	Description
Validation split handling	Done	The input data has been split for validation to improve model performance. Learn more about validation data.

+ View additional details

Type	Status	Description
Missing feature values imputation	Passed	No feature missing values were detected in the training data. Learn more about missing value imputation.

Type	Status	Description
High cardinality feature detection	Passed	Your inputs were analyzed, and no high cardinality features were detected. Learn more about high cardinality feature detection.

AutoML – Automatic Featurization

During model training with AutoML, one of the following **scaling or normalization techniques** will be applied to each model

StandardScaleWrapper — Standardize features by removing the mean and scaling to unit variance

MinMaxScalar — Transforms features by scaling each feature by that column's minimum and maximum

MaxAbsScaler —Scale each feature by its maximum absolute value

RobustScalar —Scales features by their quantile range

Principal component analysis (PCA) — Linear dimensionality reduction using Singular Value Decomposition of the data to project it to a lower dimensional space

TruncatedSVDWrapper —This transformer performs linear dimensionality reduction by means of truncated singular value decomposition (SVD). Contrary to PCA, this estimator does not center the data before computing the singular value decomposition, which means it can work with scipy.sparse matrices efficiently

SparseNormalizer —Each sample (that is, each row of the data matrix) with at least one non-zero component is rescaled independently of other samples so that its norm (l1 or l2) equals one

Feature Engineering and Selection

Training data consists of rows and columns
Each row is an observation or record, and the columns of each row are the features that describe each record

Although many of the raw data fields can be used directly to train a model t's often necessary to create additional (engineered) features that provide information that better differentiates patterns in the data

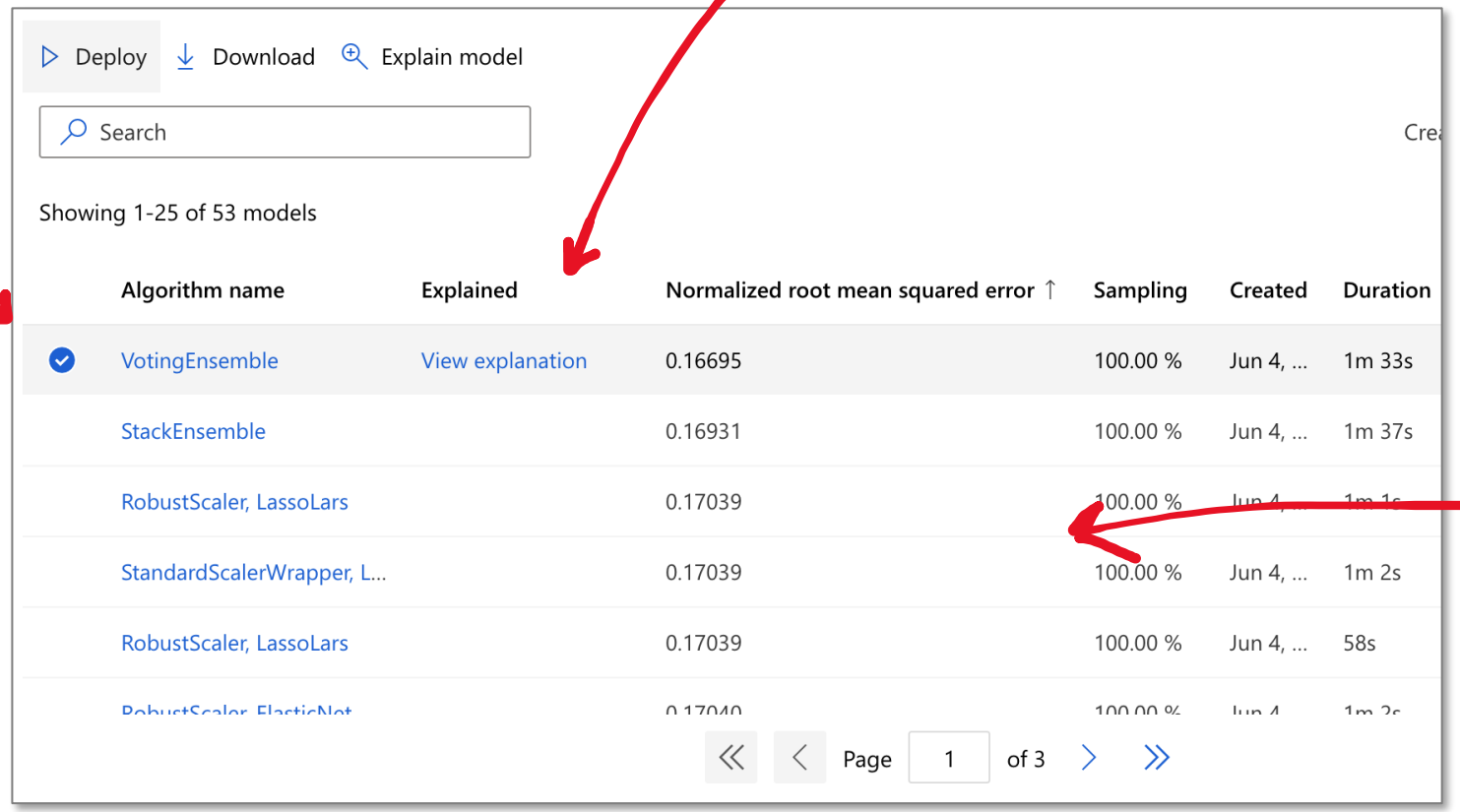
This process is called **feature engineering**, where the use of domain knowledge of the data is leveraged to create features that, in turn, help machine learning algorithms to learn better

AutoML – Model Selection

Model selection is the **task of selecting a statistical model** from a set of **candidate models**
Azure AutoML will use **many different ML Algorithms** and will recommend the best **performing candidate**

Explains the model

The top candidate



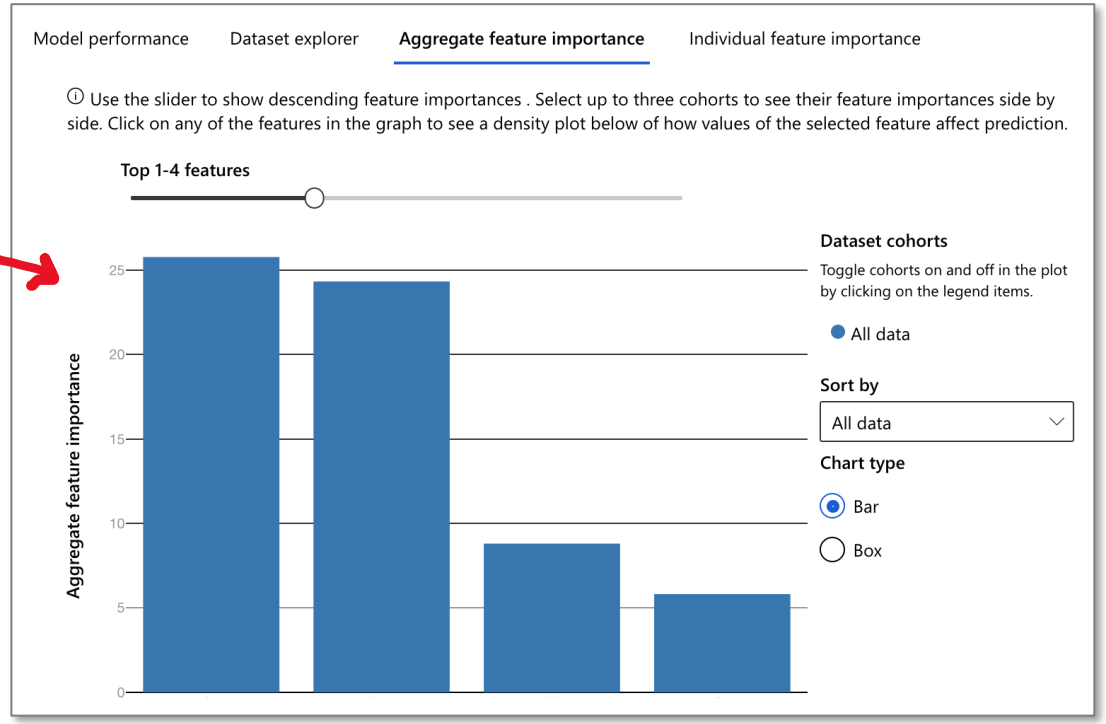
The results of running for different ML algorithms

AutoML – Explanation

ML Explainability (MLX) is the process of **explaining and interpreting** ML and deep learning models. MLX can help machine learning developers to better understand and interpret the model's behavior

After your top candidate model is selected by Azure AutoML you can get an explanation of the internals on various factors:

- Model Performance
- Dataset explorer
- **Aggerate feature importance**
- Individual feature importance



AutoML – Primary Metrics

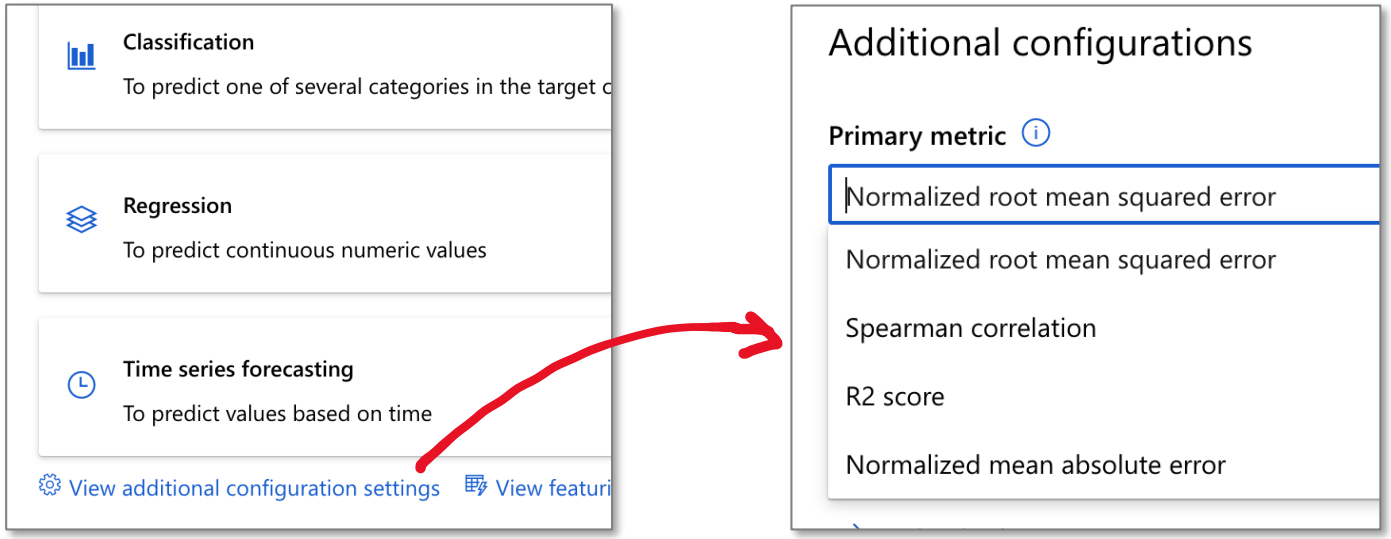
The primary metric parameter determines the metric to be used during model training for optimization.

Classification

- accuracy
- AUC_weighted
- average_precision_score_weighted
- norm_macro_recall
- precision_score_weighted

Regression and Time Series Forecasting

- spearman_correlation
- normalized_root_mean_squared_error
- r2_score
- normalized_mean_absolute_error



AutoML – Primary Metrics – Classification

Classification Scenarios

- Suited for larger datasets that well-balanced
  - **accuracy** — Image classification, Sentiment analysis, Churn prediction
  - **average_precision_score_weighted** — Sentiment analysis
  - **norm_macro_recall** — Churn prediction
  - **precision_score_weighted**
- Suited for small dataset that are imbalanced
  - **AUC_weighted** — Fraud detection, Image classification, Anomaly detection/spam detection

AutoML – Primary Metrics – Classification

Regressions Scenarios

- Works well when value to predict encompasses a large range eg. 10K to 200K
  - **spearman_correlation**
  - **r2_score** — Airline delay, Salary estimation, Bug resolution time
- Works well when value to predict encompasses as smaller range eg. 10-20K
  - **normalized_root_mean_squared_error** — Price prediction (house/product/tip), Review score prediction
  - **normalized_mean_absolute_error**

AutoML – Primary Metrics – Time Series

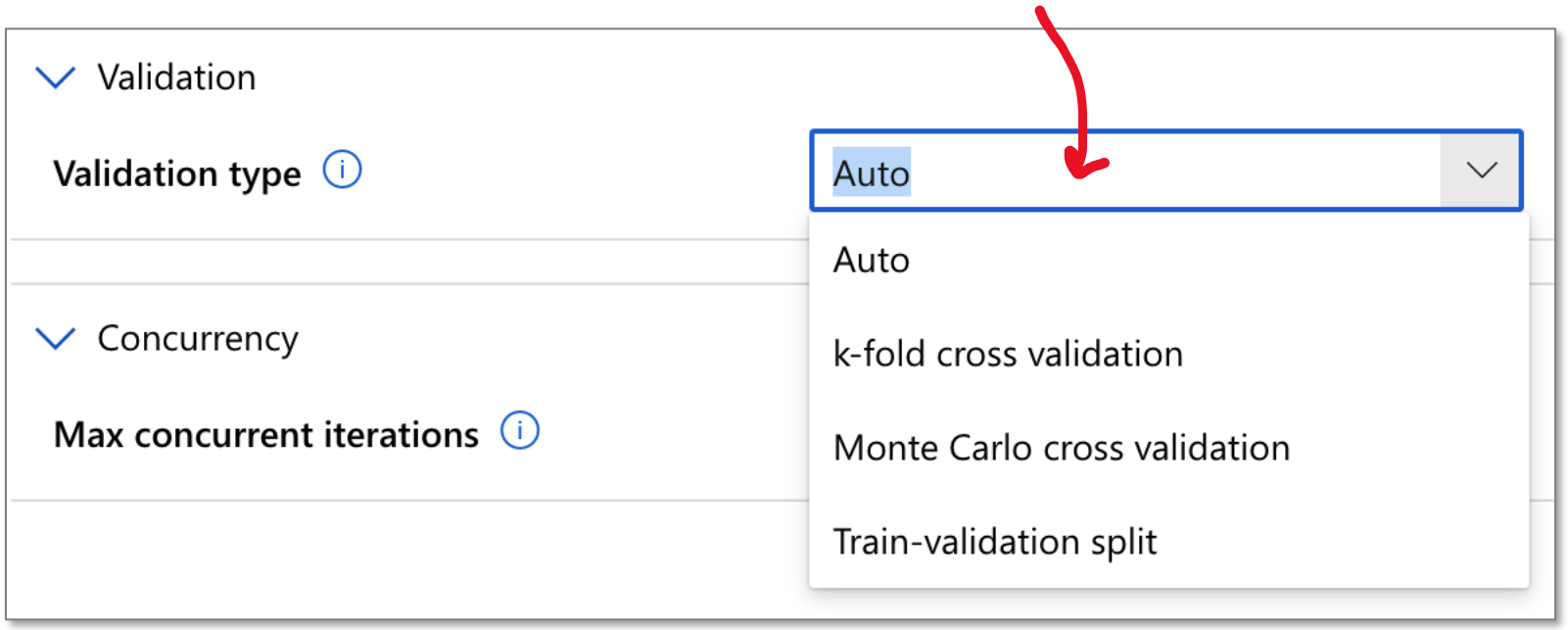
Time Series Scenarios

- Works well when value to predict encompasses a large range eg. 10K to 200K
  - **spearman_correlation**
  - **r2_score** — Price prediction (forecasting), Inventory optimization, Demand forecasting
- Works well when value to predict encompasses as smaller range eg. 10-20K
  - **normalized_root_mean_squared_error** — Price prediction (forecasting), Inventory optimization, Demand forecasting
  - **normalized_mean_absolute_error**

AutoML – Validation Type

Model Validation is when we **compare the results of our training dataset to our test dataset**.
Model Validation occurs *after* we train the model

With AutoML you can change the validation type



Custom Vision

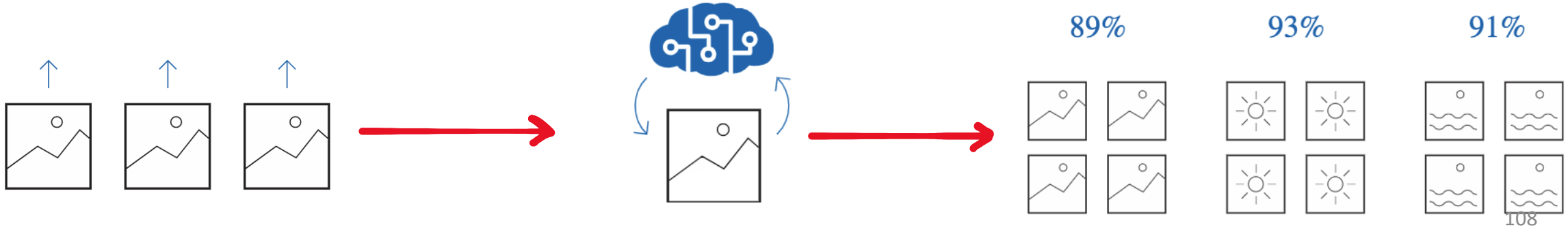
Custom Vision is a **fully-managed no-code** service to quickly build your own **Classification and Object Detection ML models**.

This service is hosted on its own isolate domain at www.customvision.ai

Upload Images
Bring your own labeled images, or use Custom Vision to quickly add tags to any unlabeled images.

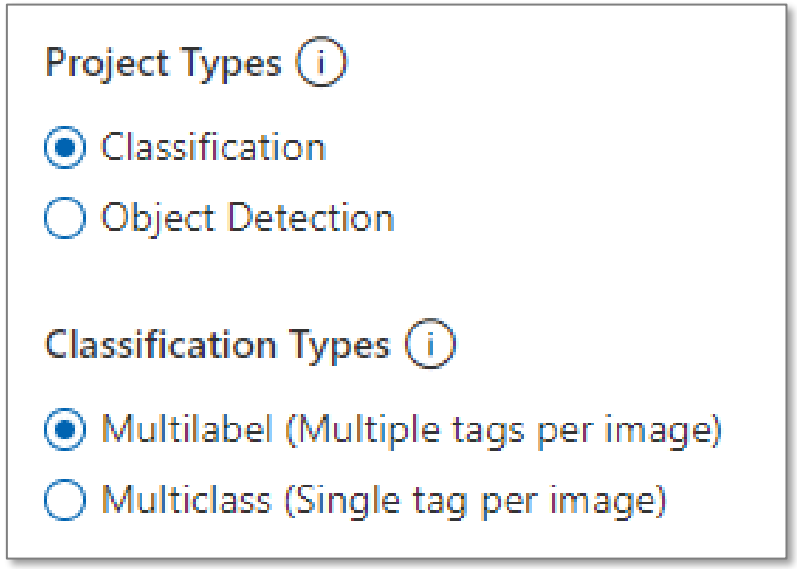
Train
Use your labeled images to teach Custom Vision the concepts you care about.

Evaluate
Use simple REST API calls to quickly tag images with your new custom computer vision model.



Custom Vision – Project Types

Within Custom Vision you setup projects and you need to select a **Project Type**



Classification

- **Multi-label**
  - When we want to apply many tags to an image
    - Image contains both a Cat and a Dog
- **Multi-class**
  - when we only have one possible tag to apply to an image:
    - It is either a Apple, Banana, Orange

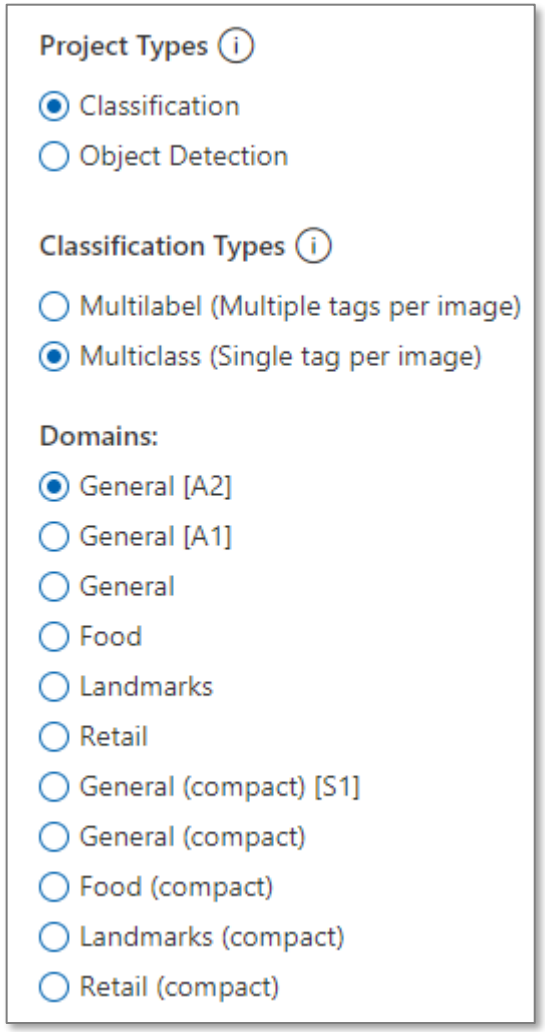
Object Detection

- When we to detect various objects in an image

You will need to also choose a **Domain**
A Domain is a Microsoft Managed dataset that is used for training the ML model
There are different domains that suited for different use cases

Custom Vision – Image Classification Domains

Classification Domains



General Optimized for a broad range of image classification tasks. If none of the other specific domains are appropriate, or if you're unsure of which domain to choose, select one of the General domains.

General [A1] Optimized for better accuracy with comparable inference time as General domain. Recommended for larger datasets or more difficult user scenarios. This domain requires more training time.

General [A2] Optimized for better accuracy with faster inference time than General[A1] and General domains. Recommended for most datasets. This domain requires less training time than General and General [A1] domains.

Food Optimized for photographs of dishes as you would see them on a restaurant menu. If you want to classify photographs of individual fruits or vegetables, use the Food domain.

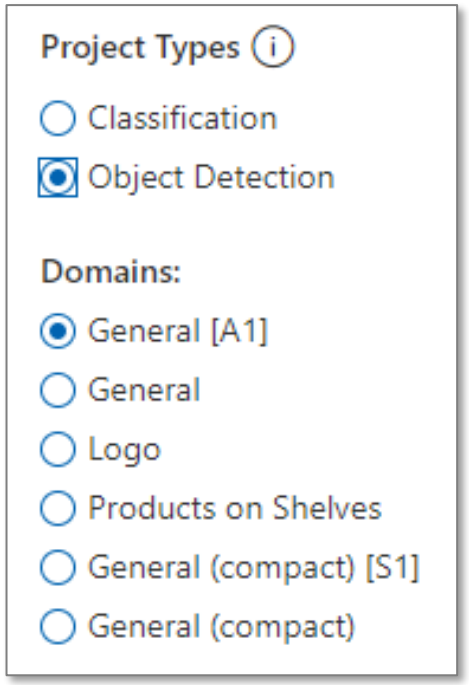
Landmarks Optimized for recognizable landmarks, both natural and artificial. This domain works best when the landmark is clearly visible in the photograph. This domain works even if the landmark is slightly obstructed by people in front of it.

Retail Optimized for images that are found in a shopping catalog or shopping website. If you want high-precision classifying between dresses, pants, and shirts, use this domain.

Compact domains Optimized for the constraints of real-time classification on edge devices.

Custom Vision – Object Detection Domains

Object Detection Domains



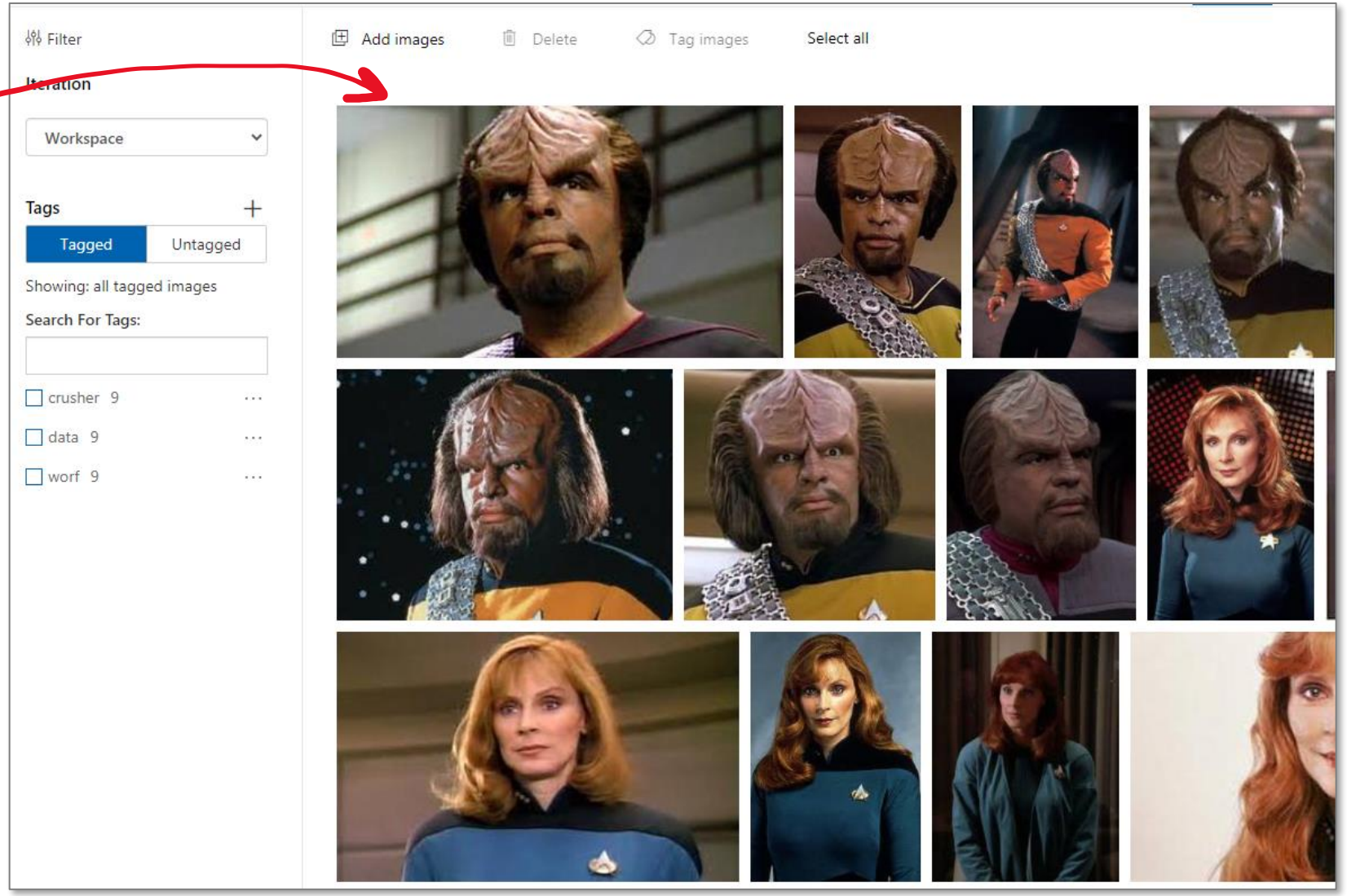
General
Optimized for a broad range of object detection tasks. If none of the other domains are appropriate, or you are unsure of which domain to choose, select the General domain.
General [A1]
Optimized for better accuracy with comparable inference time as General domain. Recommended for more accurate region location needs, larger datasets, or more difficult user scenarios. This domain requires more training time, and results are not deterministic: expect a +-1% mean Average Precision (mAP) difference with the same training data provided.

Logo
Optimized for finding brand logos in images.

Products on shelves
Optimized for detecting and classifying products on shelves.

Custom Vision – Image Classification

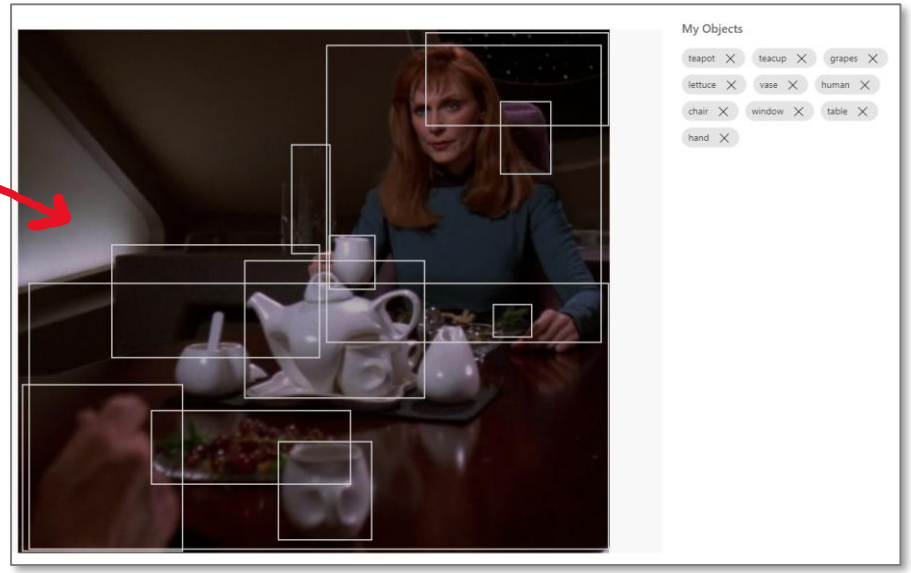
For Image Classification you **upload multiple images** and you apply a single or multiple labels to the entire image.

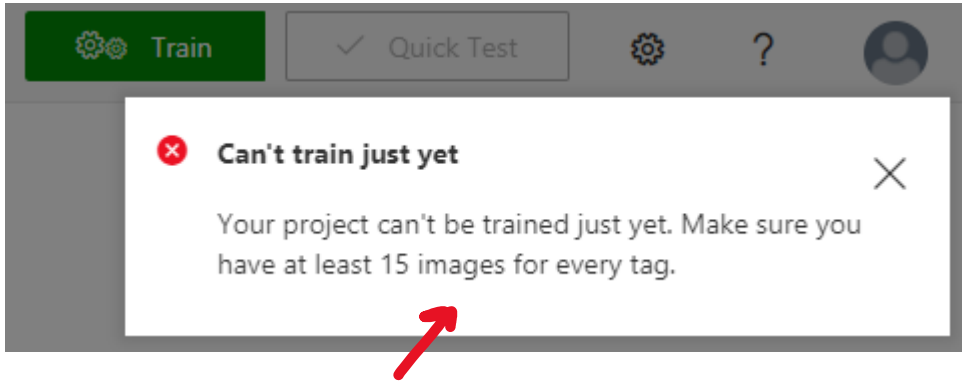


Custom Vision – Object Detection

For Object Detection you apply tags to objects in an image for data labeling

When you **hover your cursor over the image** Custom Vision uses ML to show bounding boxes of possible objects but yet have no label applied



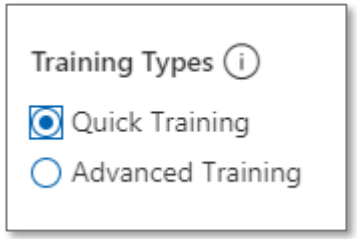


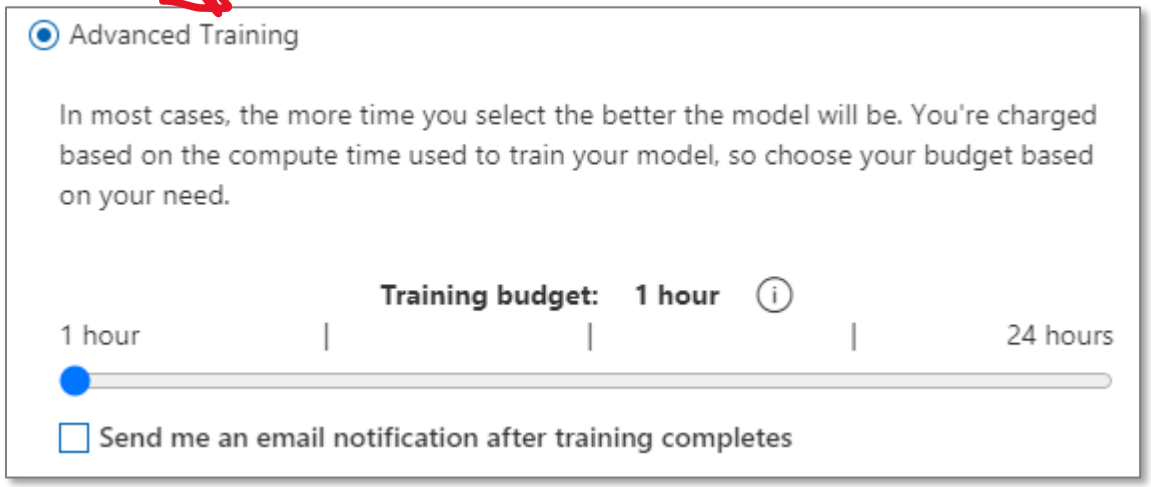
You have to **have at least 15 images** for every tag to train

Custom Vision – Training

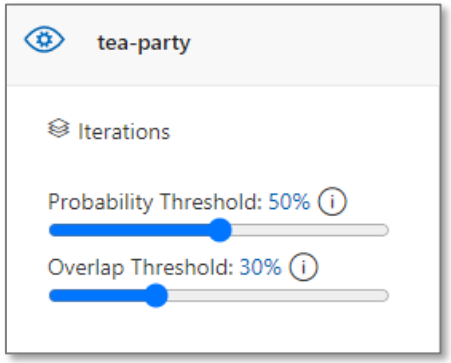
When you are ready to train you model you have two options:

- **Quick Training** – trains quickly but can be less accurate
- **Advanced Training** – increase the compute time to improve your results

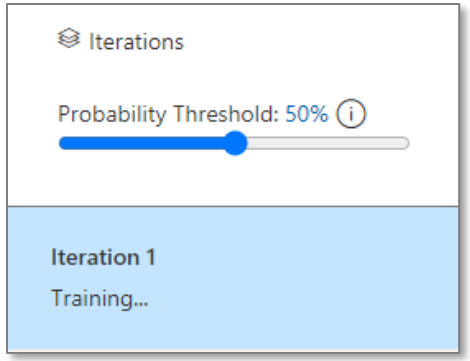




Object Detection



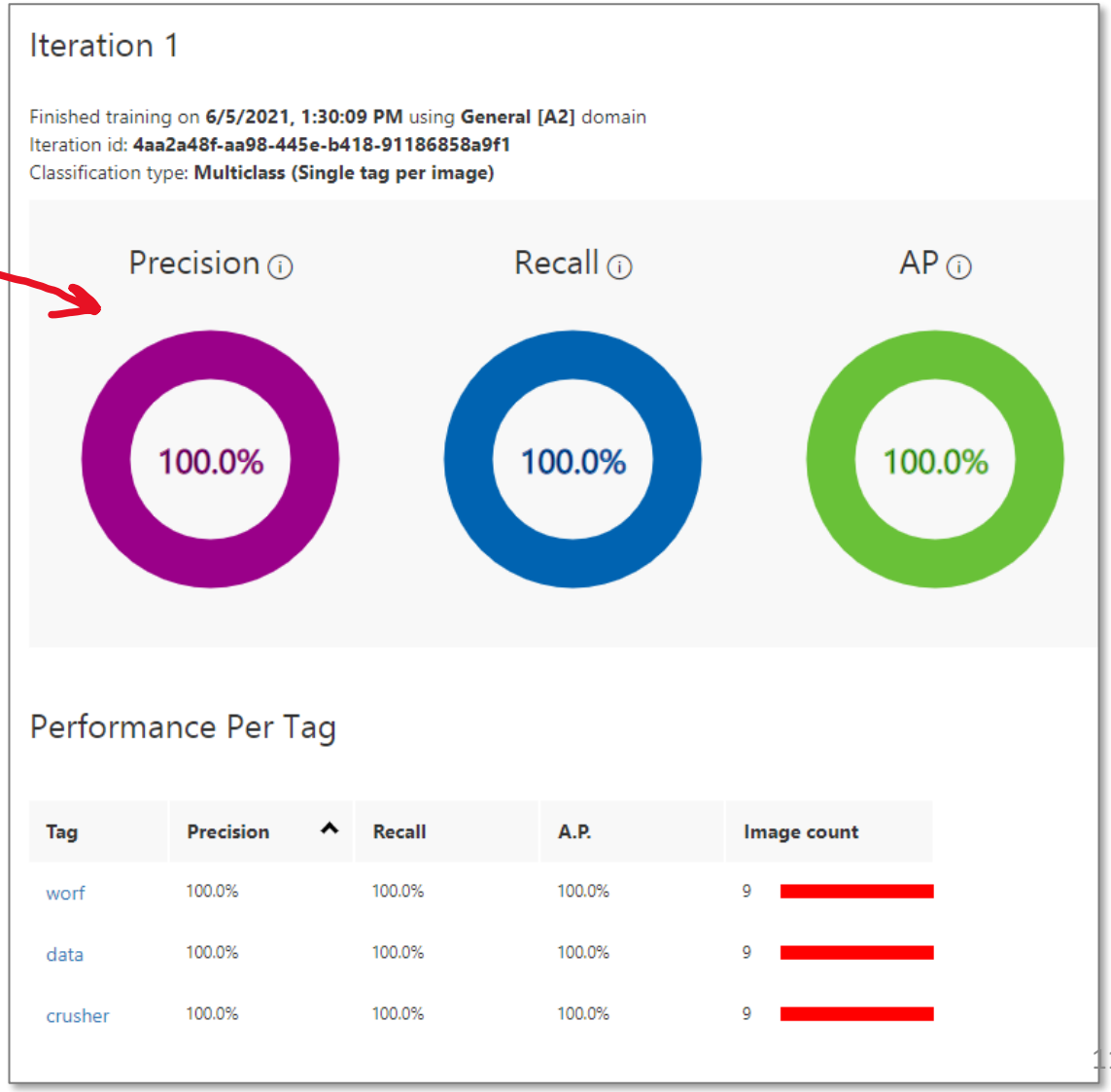
Classification



With each iteration of training our ML model will improve the evaluation metrics (**precision** and **recall**) The **probability threshold value** determines when to stop training when our evaluation metrics meet our desired threshold

Custom Vision – Training

Once the **Classification** training job is complete we will get a report of the evaluation metrics **outcome**



Precision

- being exact and accurate
- select items that are relevant

Recall (Sensitivity or True Positive Rate)

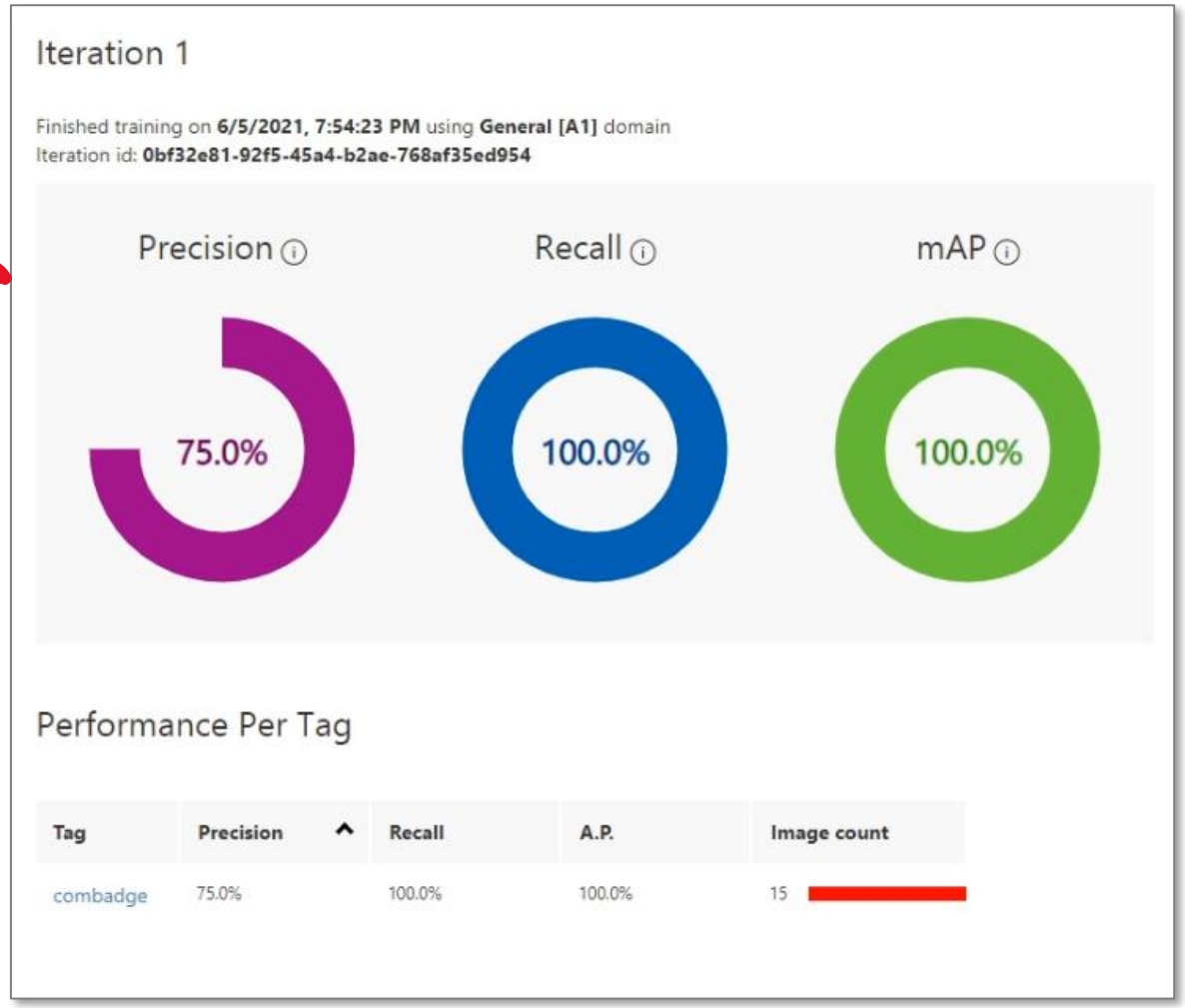
- How many relevant items returned

Average Precision (AP)

Custom Vision – Training

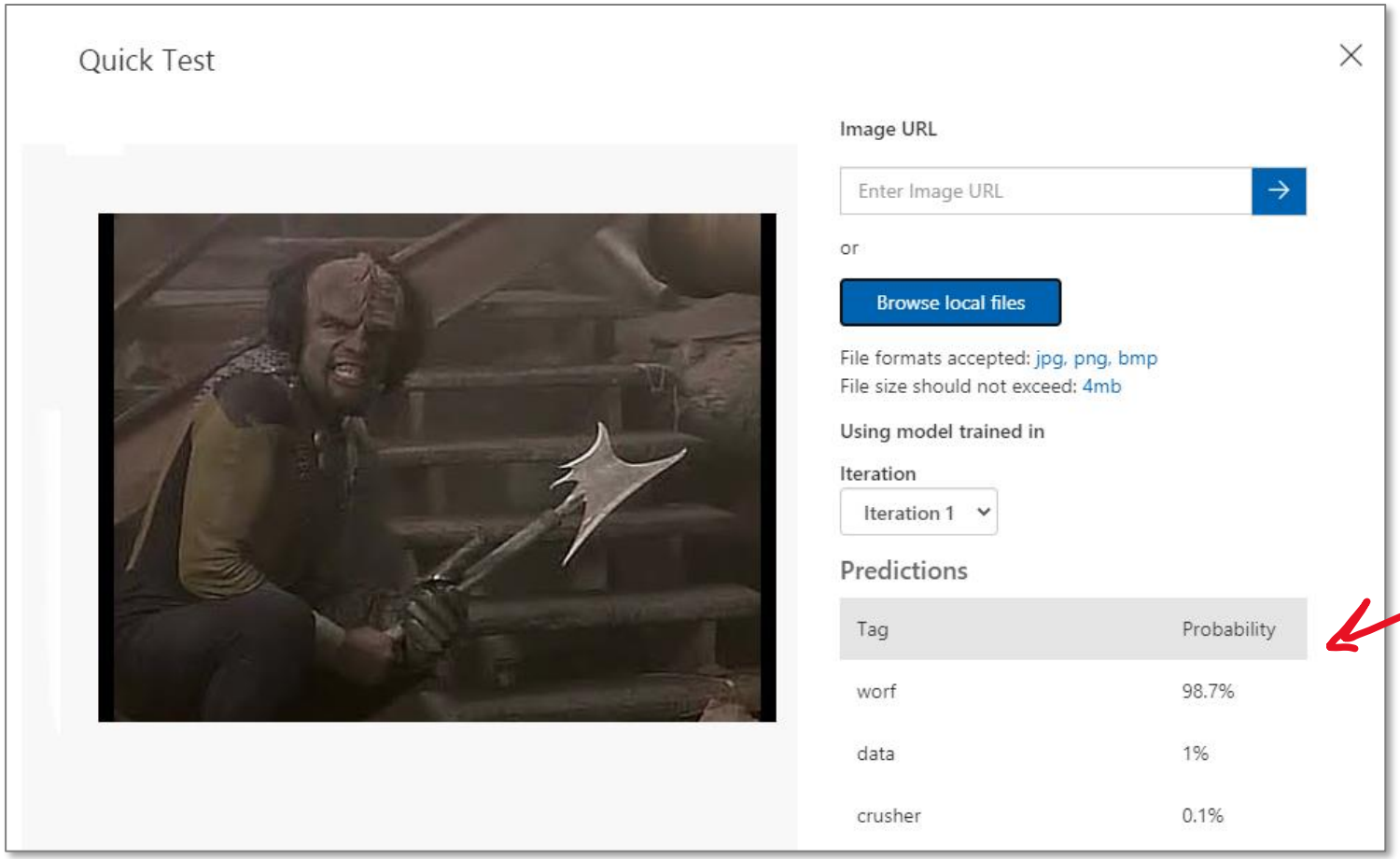
Once the **Object Dection** training job is complete we will get a report of the evaluation metrics **outcome**

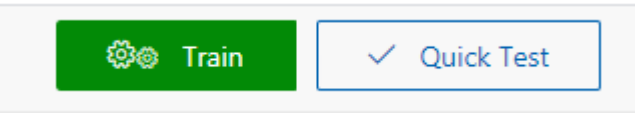
- **Precision**
- **Recall**
- **Mean Average Precision (mAP)**



Custom Vision – Quick Test

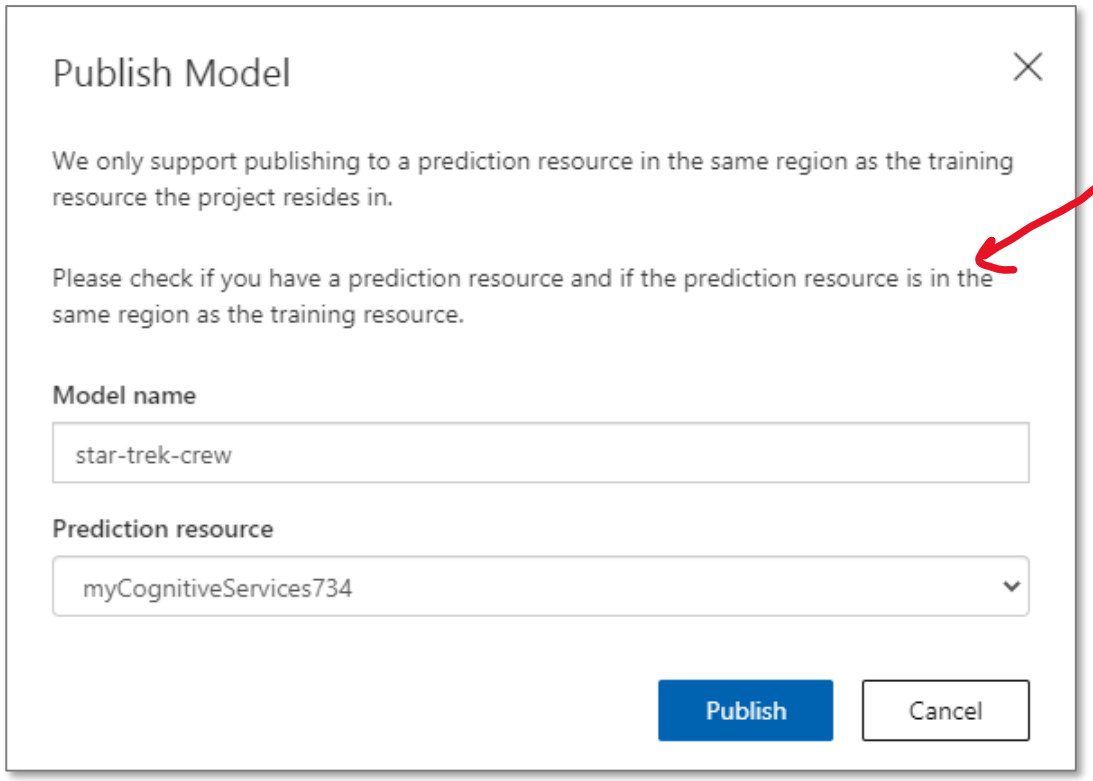
Before deploying our final trained model that can be invoked via an API Endpoint
It is good practice to test our ML model using the **Quick Test** feature



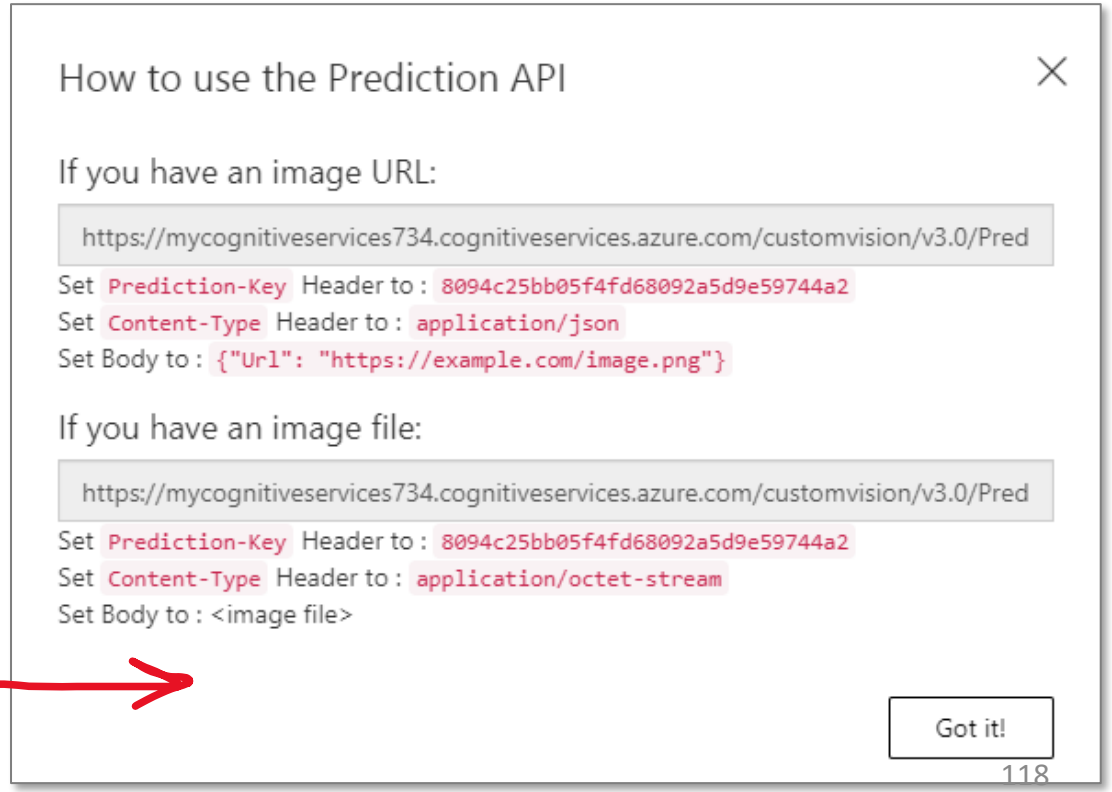


Custom Vision – Publish

To deploy our ML model to be accessible using our API Key and Endpoint we need to trigger the **Publish** action.

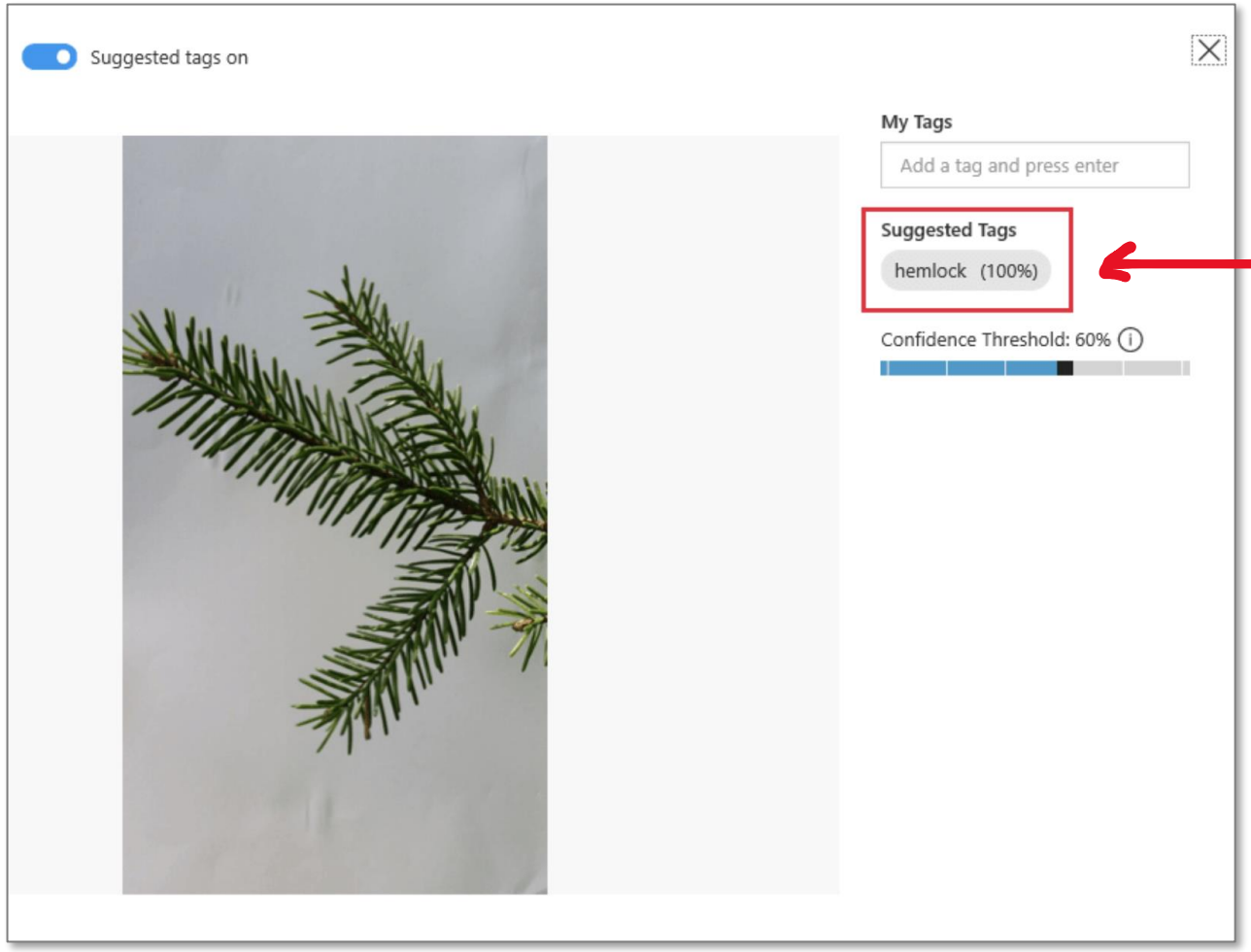


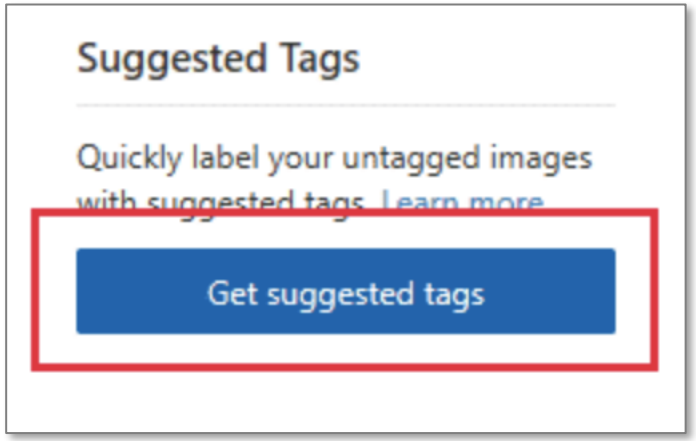
Once published we can get the **Prediction URL**



Custom Vision – Smart Labeler

When you have a very large dataset you can use Smart Labeler to predict labels.
Smart Labeler only works if you have trained the label





Smart labeler is when you want to increase your training set, and want to ML-assisted labeling to speed up this process.

Form Recognizer Service

Azure Form Recognizer is a **specialize OCR service** (translates printed text into digital and editable content) and **preserves that structure and relationship of form-like data**

Form Recognizer to **automate data entry** in your applications and **enrich your documents search capabilities**

Form Recognizer can identify:

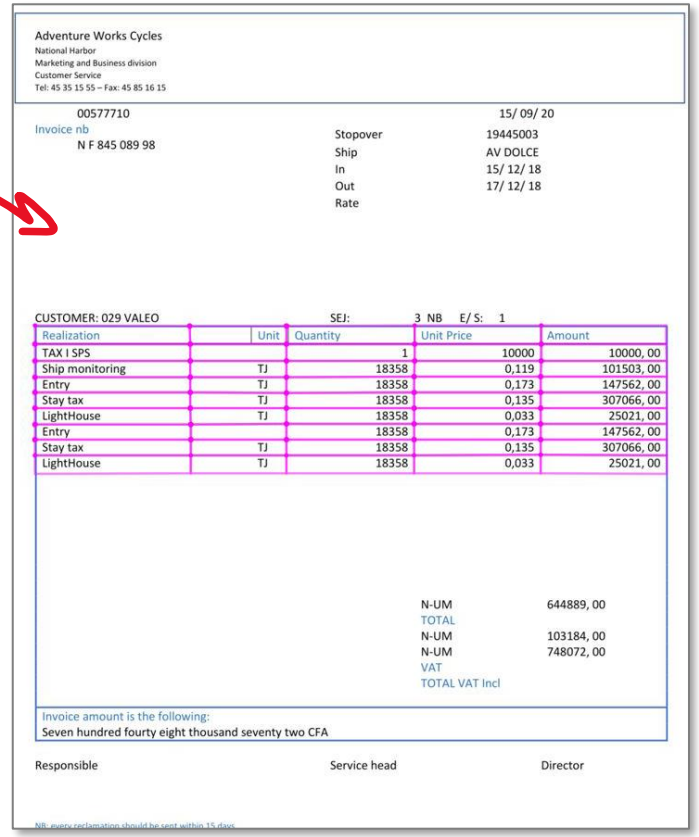
- Key Value Pairs
- Selection Marks
- Table Structures

Form Recognizer outputs structures such as:

- Original file relationships
- Bounding boxes
- Confidence scores

Form Recognizer is composed of

- Custom document processing models
- Prebuilt models for invoices, receipts, IDs, business cards
- The Layout Model



Adventure Works Cycles
National Harbor
Marketing and Business division
Customer Service
Tel: 45 35 15 55 – Fax: 45 85 16 15

00577710
Invoice nb
N F 845 089 98

15/ 09/ 20
Stopover 19445003
Ship AV DOLCE
In 15/ 12/ 18
Out 17/ 12/ 18
Rate

CUSTOMER: 029 VALEO SEJ: 3 NB E/ S: 1

Realization	Unit	Quantity	Unit Price	Amount
TAX I SPS		1	10000	10000, 00
Ship monitoring	TJ	18358	0,119	101503, 00
Entry	TJ	18358	0,173	147562, 00
Stay tax	TJ	18358	0,135	307066, 00
LightHouse	TJ	18358	0,033	25021, 00
Entry		18358	0,173	147562, 00
Stay tax	TJ	18358	0,135	307066, 00
LightHouse	TJ	18358	0,033	25021, 00

N-UM 644889, 00
TOTAL
N-UM 103184, 00
N-UM 748072, 00
VAT
TOTAL VAT Incl

Invoice amount is the following:
Seven hundred fourty eight thousand seventy two CFA

Responsible Service head Director

Form Recognizer Service – Custom Models

Custom models allows you to **extract text, key/value pairs, selection marks, and table data** from forms

- These models are trained with your own data, so they're tailored to your forms
- you only need five sample input forms to start.
- A trained document processing model can output structured data that includes the relationships in the original form document.
- After you train the model, you can test and retrain it and eventually use it to reliably extract data from more forms according to your needs.

You have **2 learning options**:

Train without Labels
uses **unsupervised learning** to understand the layout and relationships between fields and entries in your forms

Train with Labels
uses **supervised learning** to extract values of interest, using the labeled forms you provide (trained data).

Form Recognizer Service – Prebuilt Models

Receipts

Sales receipts from Australia, Canada, Great Britain, India, and the United States

Alamo Drafthouse Cinema
www.originalalamo.com
1120 South Lamar
Austin, TX 78704
512-707-8262

Server: [redacted] 05/07/2009
Table 1512/1 8:04 PM
Guests: 1 20084

SM Water	0.00
earl gray tea	2.50
Romulan Ale	5.50
klingon wine	5.50
Royale w/chz	8.99
Sub Total	22 49
Tax	1.86
Total	24.35
Balance Due	**24.35**

Thanks for coming in!
Comments? E-mail us at
comments.lamar
@originalalamo.com

Fields Extracted

ReceiptType
MerchantName
MerchantPhoneNumber
MerchantAddress
TransactionDate
TransactionTime
Total
Subtotal
Tax
Tip
Items
Name
Quantity
Price
Total Price

Form Recognizer Service – Prebuilt Models

Business Cards
English business cards

Fields Extracted
ContactNames
FirstName
LastName
CompanyNames
Departments
JobTitles
Emails
Websites
Addresses
MobilePhones
Faxes
WorkPhones
OtherPhones

Form Recognizer Service – Prebuilt Models

Invoices

extracts data from invoices in various formats and returns structured data

Extracted Fields

CustomerName
CustomerId
PurchaseOrder
InvoiceId
InvoiceDate
DueDate
VendorName
VendorAddress
VendorAddressRecipient
CustomerAddress
CustomerAddressRecipient
BillingAddress
BillingAddressRecipient
ShippingAddress
ShippingAddressRecipient
SubTotal
TotalTax
InvoiceTotal
AmountDue
ServiceAddress
ServiceAddressRecipient
RemittanceAddress
RemittanceAddressRecipient
ServiceStartDate
ServiceEndDate
PreviousUnpaidBalance

Extracted Line Item Data

Items
Amount
Description
Quantity
UnitPrice
ProductCode
Unit
Date
Tax

Form Recognizer Service – Prebuilt Models

Identity Documents (IDs)
world-wide passports and US driver licenses

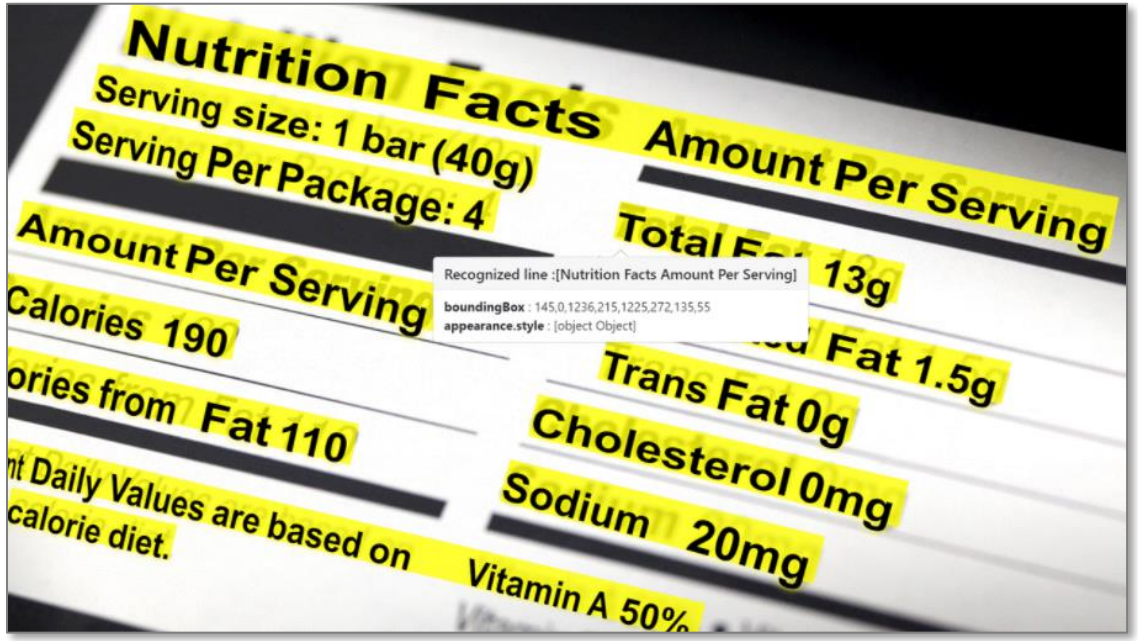
Extracted Fields
CountryRegion
DateOfBirth
DateOfExpiration
DocumentNumber
FirstName
LastName
Nationality
Sex
MachineReadableZone
DocumentType
Address
Region

Optical Character Recognition (OCR)

Optical character recognition (OCR) is the process of **extracting printed or handwritten text into a digital and editable format**

OCR can be applied to:

- photos of street signs
- **Products**
- Documents
- Invoices
- Bills
- Financial Reports
- Articles
- and more



Optical Character Recognition (OCR)

Azure has **two different APIs** that can perform OCR: **OCR API** and **Read API**

OCR API

- older recognition model
- supports only images
- executes synchronously
  - returning immediately with the detected text
  - Suited for less text
- Support more languages
- Easier to implement

Read API

- updated recognition model
- Supports images and PDFs
- Executes asynchronously
  - parallelizes tasks per line for faster results
  - Suited for lots of text
- Supports fewer languages
- A bit more difficult to implement

OCR is performed via the **Computer Vision SDK**

AI vs Generative AI

Artificial Intelligence (AI)

AI refers to the development of computer systems that can **perform tasks typically requiring human intelligence**. These include **problem-solving, decision-making, understanding natural language, recognizing speech and images**, and more.

The primary goal of traditional AI is to create systems that can **interpret, analyze, and respond to human actions** or environmental changes efficiently and accurately. It aims to replicate or simulate human intelligence in machines.

AI applications are vast and include areas like **expert systems, natural language processing, speech recognition, and robotics**.

AI is used in various industries for tasks such as **customer service chatbots, recommendation systems in e-commerce, autonomous vehicles, and medical diagnosis**.

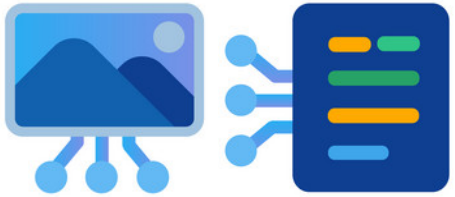
AI vs Generative AI

Generative AI

Generative AI is a subset of AI that focuses on **creating new content or data** that is novel and realistic. It does not just interpret or analyze data but **generates new data itself**.
It includes **generating text, images, music, speech, and other forms of media**.

It often involves advanced machine learning techniques, particularly deep learning models like **Generative Adversarial Networks (GANs), Variational Autoencoders (VAEs), and Transformer models (like GPT).**

Generative AI is used in a range of applications including creating realistic **images and videos**, generating **human-like text**, composing **music**, creating virtual environments, and even drug discovery.



Examples: Tools like **GPT (Generative Pre-trained Transformer)** for text generation, **DALL-E** for image creation, and various deep learning models that compose music.

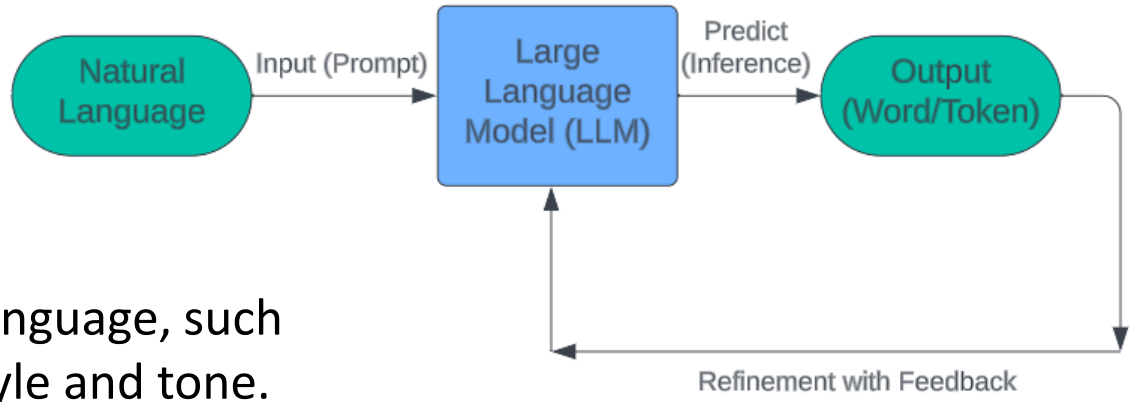
AI vs Generative AI

Feature	Artificial Intelligence (AI)	Generative AI
Functionality	Regular AI focuses on understanding and decision-making	Generative AI is about creating new, original outputs.
Data Handling	AI typically analyzes and makes decisions based on existing data	Generative AI uses existing data to generate new, unseen outputs.
Applications	Its applications span across various sectors, including data analysis, automation, natural language processing, and healthcare.	Its applications are more creative and innovative, focusing on content creation, synthetic data generation, deepfakes, and design.

What is a Large Language Model (LLM)?

A Large Language Model (LLM) such as GPT (Generative Pre-trained Transformer) works in a way that's similar to a complex, **automatic system that recognizes patterns and makes predictions**.

Training on Large Datasets: Initially, the model is trained on massive amounts of text data. This data can include **books, articles, websites, and other written material**.



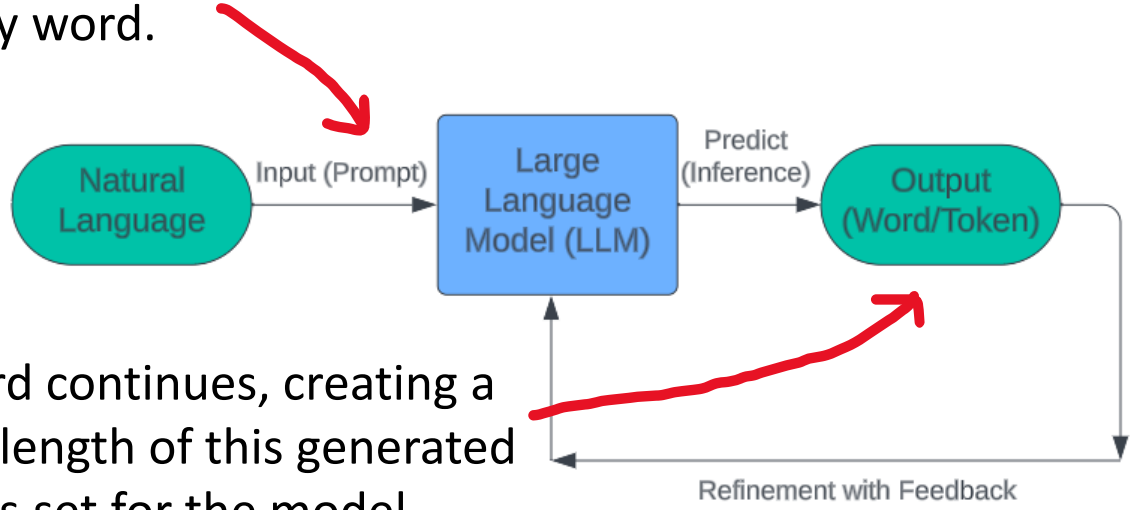
During this training phase, the model learns patterns in language, such as grammar, word usage, sentence structure, and even style and tone.

Understanding Context: The model's design allows it to consider a wide context. This means it doesn't just focus on single words, but understands them in **relation to the words and sentences** that come **before and after**. This context understanding is important for generating coherent and relevant text.

What is a Large Language Model (LLM)?

Predicting the Next Word: When you give the model a **prompt** (a starting piece of text), it uses what it has learned to predict the next most likely word.

It then adds this word to the prompt and repeats the process, continually **predicting the next word** based on the extended sequence.



Generating Text: This process of predicting the next word continues, creating a **chain of words** that forms a coherent piece of text. The length of this generated text can vary based on specific instructions or limitations set for the model.

Refinement with Feedback: The model can be further **refined** and **improved** over time with **feedback**. This means it gets better at understanding and generating text as it is exposed to more data and usage.

Transformer models

A transformer model is a type of machine learning model that's especially good at **understanding and generating language**.

It's built using a structure called the transformer architecture, which is really effective for tasks involving **natural language processing (NLP)**, like **translating languages or writing text**.

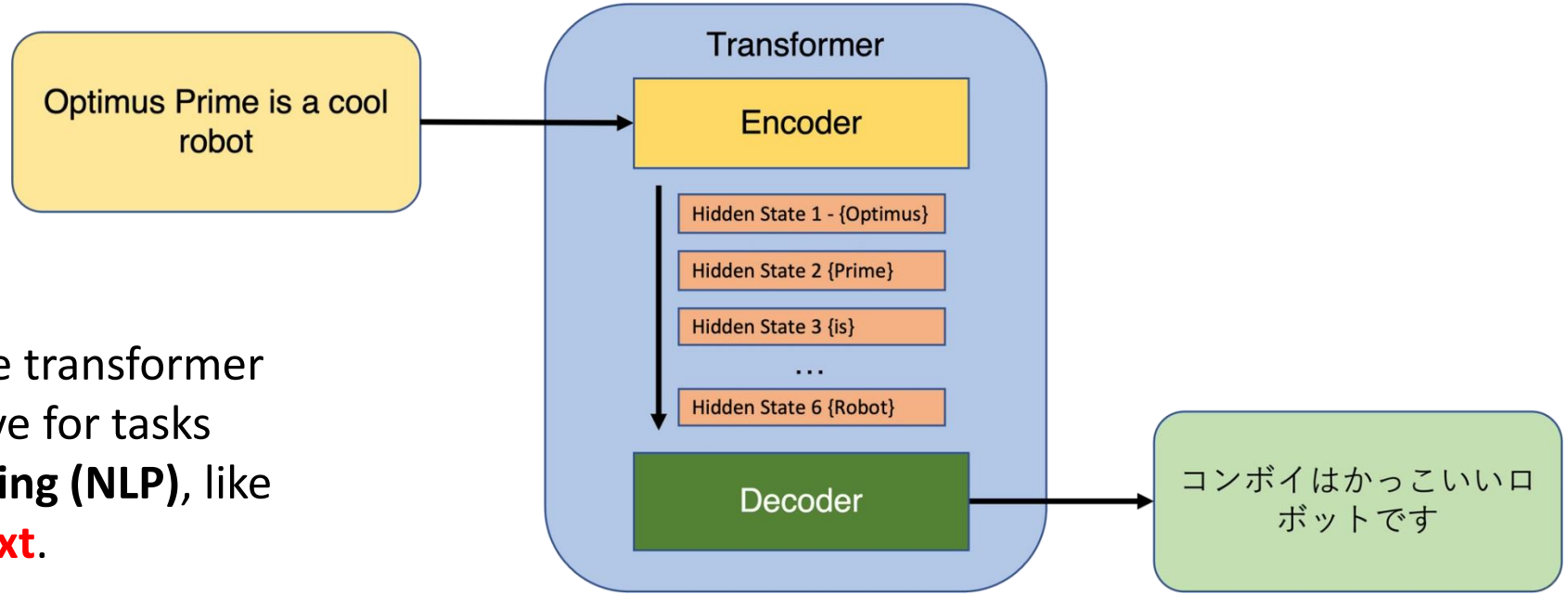


Image credits: domino.ai

Transformer model architecture consists of **two components**, or *blocks*:

1. Encoder: This part **reads and understands the input text**. It's like a smart system that goes through everything it's been taught (which is a lot of text) and picks up on the meanings of words and how they're used in different contexts.

2. Decoder: Based on what the encoder has learned, this part **generates new pieces of text**. It's like a skilled writer that can make up sentences that flow well and make sense.

Transformer models

There are different types of transformer models with specific jobs. For example:

BERT is good at understanding the language. It's like a librarian who knows where every book is and what's inside them. **Google** uses it to help its search engine understand what you're looking for.

GPT is good at creating text. It's like a skilled author who can write stories, articles, or conversations based on what it has learned.

Tokenization

Tokenization in a transformer model is like turning a sentence into a puzzle. For example, you have the sentence: **"I heard a dog bark loudly at a cat."** To help a computer understand it, we chop up the sentence into pieces called **'tokens'**. Each piece can be a word or even a part of a word.
So, for our sentence, we give each word a number, like this:

- "I" might be 1
- "heard" might be 2
- "a" might be 3
- "dog" might be 4
- "bark" might be 5
- "loudly" might be 6
- "at" might be 7
- *"a" is already tokenized as 3*
- "cat" might be 8

Tokenization: Turning words into tokens

Now, our sentence becomes a series of numbers: **[1, 2, 3, 4, 5, 6, 7, 3, 8].** This is like giving each word a **special code**.
The computer uses these codes to **learn about the words and how they fit together**.
If a word repeats, like **"a"**, we use its code again instead of making a new one.

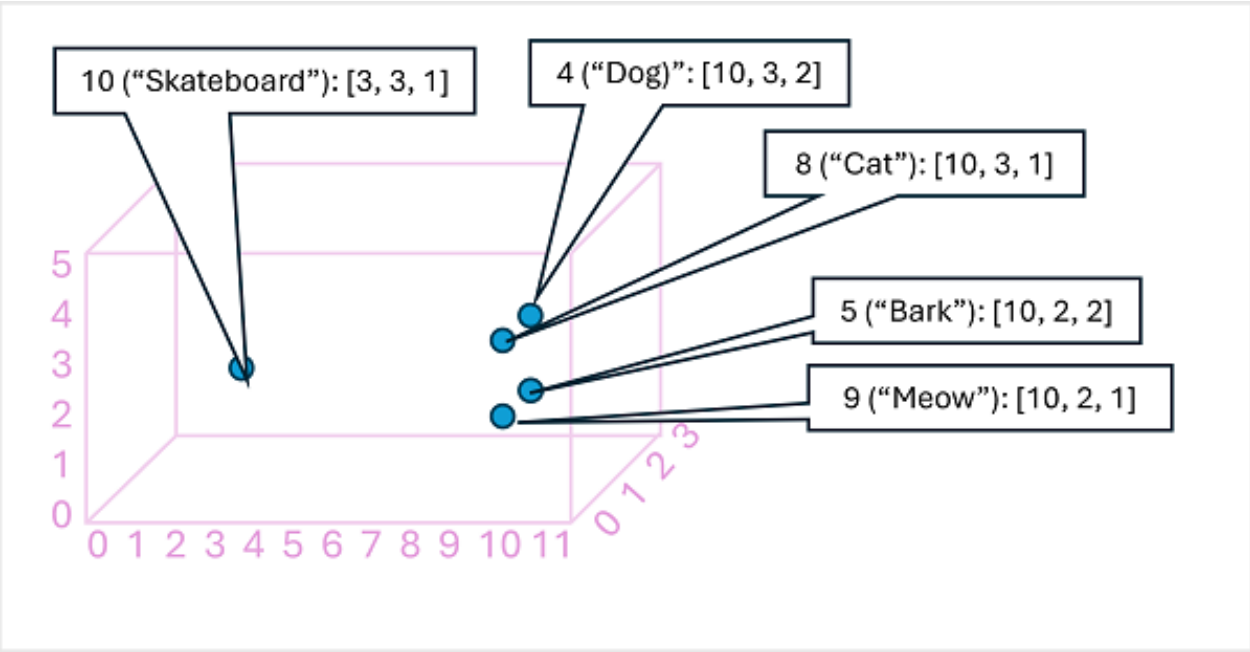
- As the computer reads more text, it keeps turning new words into new tokens with new numbers.
- If it learns the word "meow," it might call it 9, and "skateboard" could be 10.

Embeddings

To help a computer understand language, we turn words into tokens and then give each token a special **numeric code**, called an **embedding**. These embeddings are like a secret code that captures the meaning of the word.
As a simple example, suppose the embeddings for our tokens consist of **vectors** with three elements, for example:

- 4 ("dog"): [10,3,2]
- 5 ("bark"): [10,2,2]
- 8 ("cat"): [10,3,1]
- 9 ("meow"): [10,2,1]
- 10 ("skateboard"): [3,3,1]

Words that have **similar meanings** or are used in similar ways get **codes that look alike**.
So, "dog" and "bark" might have similar codes because they are **related**.



Embedding: Turning words (tokens) into vectors (lists of numbers)

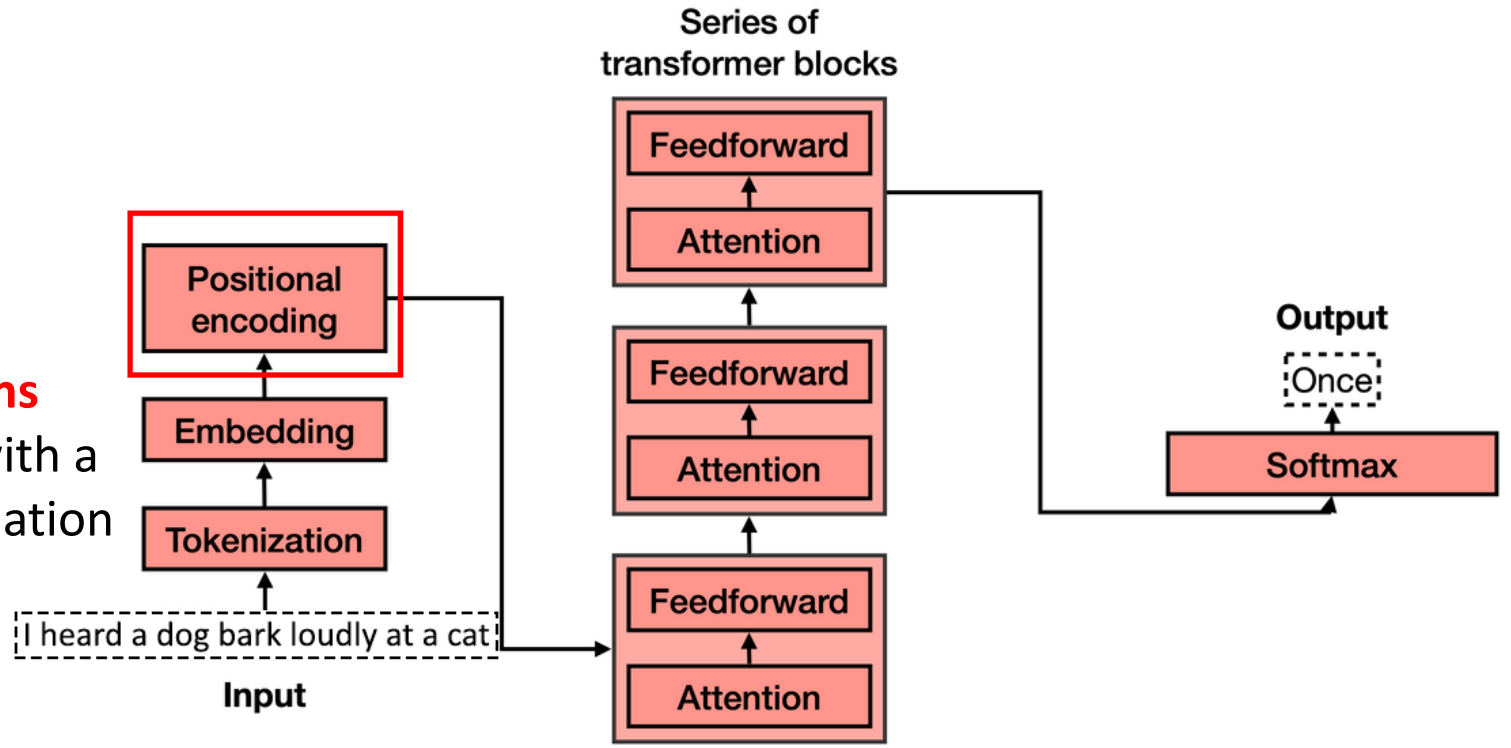
This way, the computer can figure out which words are **similar to each other** just by looking at their codes.
It's like giving each word a home on a map, and words that are neighbors on this map have related meanings.

Positional encoding

Positional encoding is a technique used to ensure that a language model, such as GPT (Generative Pre-trained Transformer) doesn't lose the **order of words** when processing natural language. This is important because the order in which words appear can change the meaning of a sentence.

Let's take the sentence **"I heard a dog bark loudly at a cat"** from our previous example:

Without positional encoding, if we simply tokenize this sentence and convert the **tokens** into **embedding vectors**, we might end up with a set of vectors that **lose the sequence** information



Positional encoding adds a positional vector to each word to keep track of the positions of the words.

Positional encoding

However, by adding **positional encoding vectors to each word's embedding**, we ensure that each **position** in the sentence is uniquely identified:

- The embedding for "I" would be modified by adding a positional vector corresponding to position 1, labeled **"I (1)".**
- The embedding for "heard" would be altered by a vector for position 2, labeled **"heard (2)"**.
- The embedding for "a" would be updated with a vector for position 3, labeled **"a (3)"**, and reused with the same positional vector for its second occurrence.
- This process continues for each word/token in the sentence, with **"dog (4)", "bark (5)", "loudly (6)", "at (7)", and "cat (8)"** all receiving their unique positional encodings.

As a result, the sentence **"I heard a dog bark loudly at a cat"** is represented not just by a sequence of vectors for its words, but by a sequence of vectors that are influenced by the **position** of each word in the sentence.

This means that even if another sentence had the same words in a different order, its overall representation would be different because the positional encodings would differ, reflecting the different sequence of words.

Attention

Attention in AI, especially in transformer models, is a way the model figures out how **important each word (or token) is to the meaning of a sentence**, particularly in **relation** to the other words around it. Let's reuse the sentence **"I heard a dog bark loudly at a cat"** to explain this better:

1. **Self-Attention**: Imagine each word in the sentence shining a flashlight on the other words. The brightness of the light shows how much one word should pay attention to the others when understanding the sentence.
   For **"bark",** the light might shine brightest on **"dog"** because they're closely related.

2. **Encoder's Role**: In the encoder part of a transformer model, attention helps decide **how to represent each word as a number (or vector)**. It's not just the word itself, but also its context that matters.
   For example, "bark" in **"the bark of a tree"** would have a different representation than "bark" in **"I heard a dog bark",** because the surrounding words are different.

Attention

3 **Decoder's Role**: When generating new text, like completing a sentence, the decoder uses attention to figure out which words it already has are **most important** for **deciding what comes next**. If our sentence is "I heard a dog," the model uses **attention** to know that **"heard"** and **"dog"** are key to adding the next word, which might be **"bark."**

4 **Multi-Head Attention**: It's like having multiple flashlights, each **highlighting different aspects of the words**. Maybe one flashlight looks at the **meaning** of the word, another looks at its **role** in the sentence (like subject or object), and so on. This helps the model get a richer understanding of the text.

5 **Building the Output**: The decoder builds the sentence one word at a time, using **attention** at each step. It looks at the sentence so far, decides what's important, and then **predicts** the next word. It's an ongoing process, with each **new word influencing the next**.

Attention in transformer models is like a guide that helps the AI understand and create language by focusing on the most relevant parts of the text, considering both individual word meanings and their relationships within the sentence.

Attention – Process

1 **Token Embeddings**: Each word in the sentence is represented as a **vector** of numbers (its **embedding**).

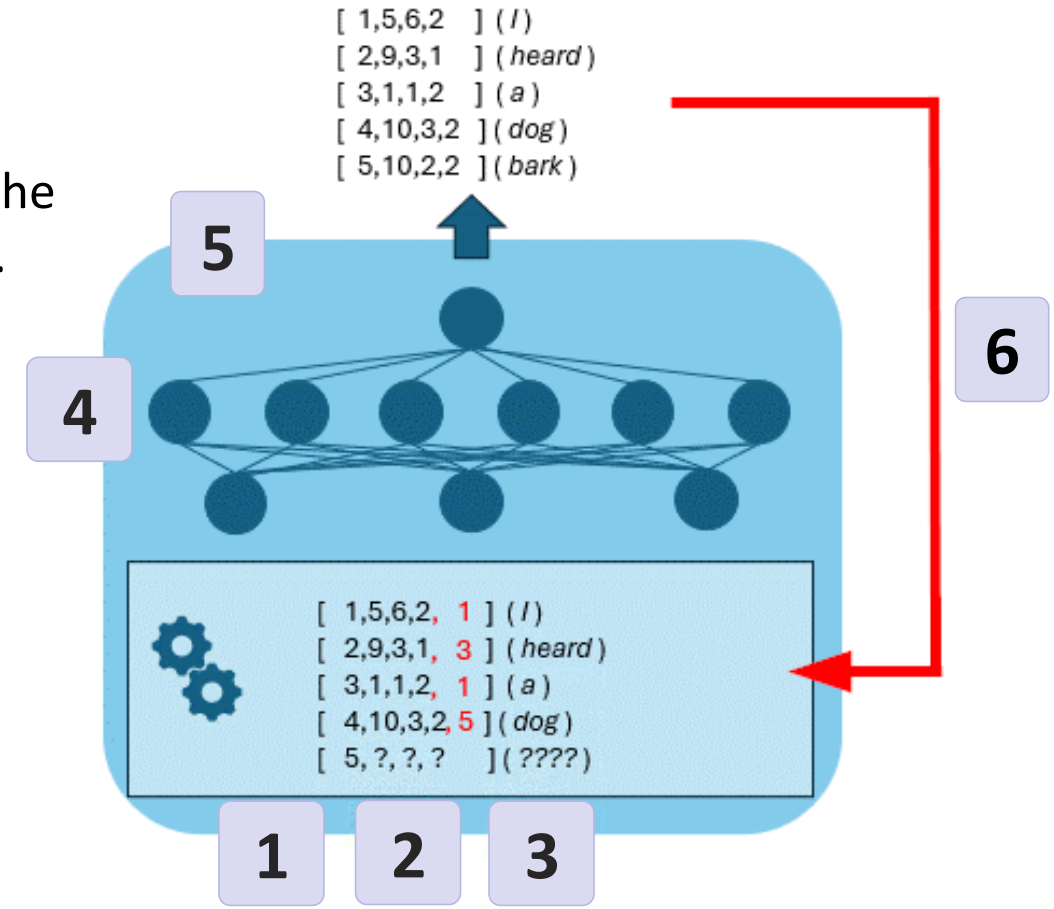
2 **Predicting the Next Token**: The goal is to figure out what the next **word (token)** should be, also represented as a vector.

3 **Assigning Weights**: The attention layer looks at the sentence so far and decides how much **influence (weight)** each word should have on the next one.

4 **Calculating Attention Scores**: Using these weights, a new vector for the next token is calculated, which includes an attention score. **Multi-head attention** does this several times, focusing on different aspects of the words.

5 **Choosing the Most Likely Word**: A neural network takes these vectors with **attention scores** and picks the **word** from the vocabulary that most likely comes next.

6 **Adding to the Sequence**: The chosen word is **added to the existing sequence**, and the process **repeats for each new word**.



GPT-4

A transformer model like **GPT-4** works by taking a text **input (prompt)** and producing a well-structured **output (completion)**. During training, it learns from a vast array of text data, understanding how words are typically arranged in sentences.

The model knows the correct sequence of words but **hides (masks)** future words to learn how to **predict** them. When it tries to predict a word, it compares its guess to the actual word, gradually adjusting to reduce errors.

In practice, the model uses its training to assign **importance (weights)** to each word in a sequence, helping it guess the next word accurately. The result is that GPT-4 can create **sentences** that sound like they were written by a human.

However, this doesn't mean the model "knows" things or is "intelligent" in the human sense. It's simply very good at using its **large vocabulary and training to generate realistic text based on word relationships.**

Azure OpenAI Service

Azure OpenAI Service is a cloud-based platform designed to **deploy and manage advanced language models from OpenAI**. This service combines OpenAI's latest language model developments with the robust security and scalability of Azure's cloud infrastructure.

Azure OpenAI offers several **types of models** for different purposes:

- **GPT-4 Models**: These are the newest in the line of GPT models and can create **text and programming code** when given a prompt written in natural language.
- **GPT-3.5 Models**: Similar to GPT-4, these models also create text and code from natural language prompts. The GPT-3.5-turbo version is specially designed for **conversations**, making it a great choice for **chat applications and other interactive AI tasks**.
- **Embedding Models**: These models turn written **text into number sequences**, which is helpful for analyzing and comparing different pieces of text to find out how **similar** they are.
- **DALL-E Models**: These models can make **images from descriptions** given in words. The DALL-E models are still being tested and aren't shown in the Azure OpenAI Studio, so you don't have to set them up for use manually.

Azure OpenAI Service

Key concepts in using Azure OpenAI include **prompts and completions, tokens, resources, deployments, prompt engineering, and various models**:

Prompts & Completions: Users interact with the **API** by providing a text command in English, known as a **prompt**, and the model generates a text response, or completion.

- E.g., a prompt to count to five in a loop results in the model returning appropriate code.

Tokens: Azure OpenAI breaks down text into **tokens**, which are words or character chunks, to process requests. The number of tokens affects response latency and throughput.

- For images, token cost varies with image size and detail setting, with low-detail images costing fewer tokens and high-detail images costing more.

Resources: Azure OpenAI operates like other **Azure products** where users create a resource within their Azure Subscription.

Deployments: To use the service, users must deploy a **model via Deployment APIs**, choosing the specific model for their needs.

Prompt Engineering: Crafting **prompts** is crucial as they guide the model's output.
This requires skill, as prompt construction is nuanced and impacts the model's response.

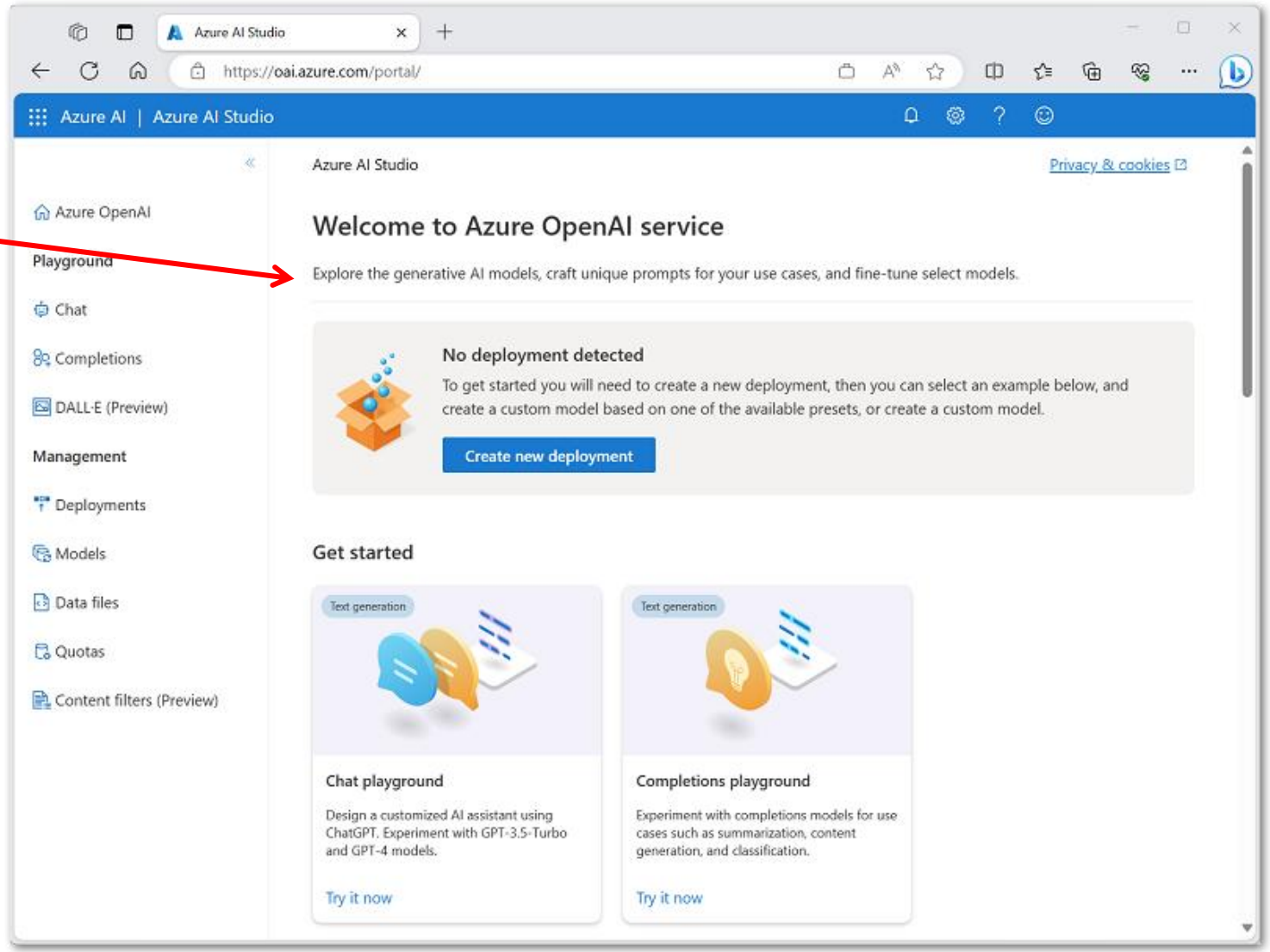
Models: Various models offer different capabilities and pricing. **DALL-E** creates images from text, while **Whisper** transcribes and translates speech to text. Each has unique features suitable for different tasks.

Azure OpenAI Studio

Developers can work with these models in **Azure OpenAI Studio**, a **web-based environment** where AI professionals can **deploy, test, and manage LLMs** that support generative AI app development on Azure.

Access is currently **limited** due to the high demand, upcoming product improvements, and Microsoft's commitment to responsible AI.

Presently, collaborations are being prioritized for those who already have a **partnership with Microsoft**, are engaged in lower-risk use cases, and are dedicated to including necessary safeguards.



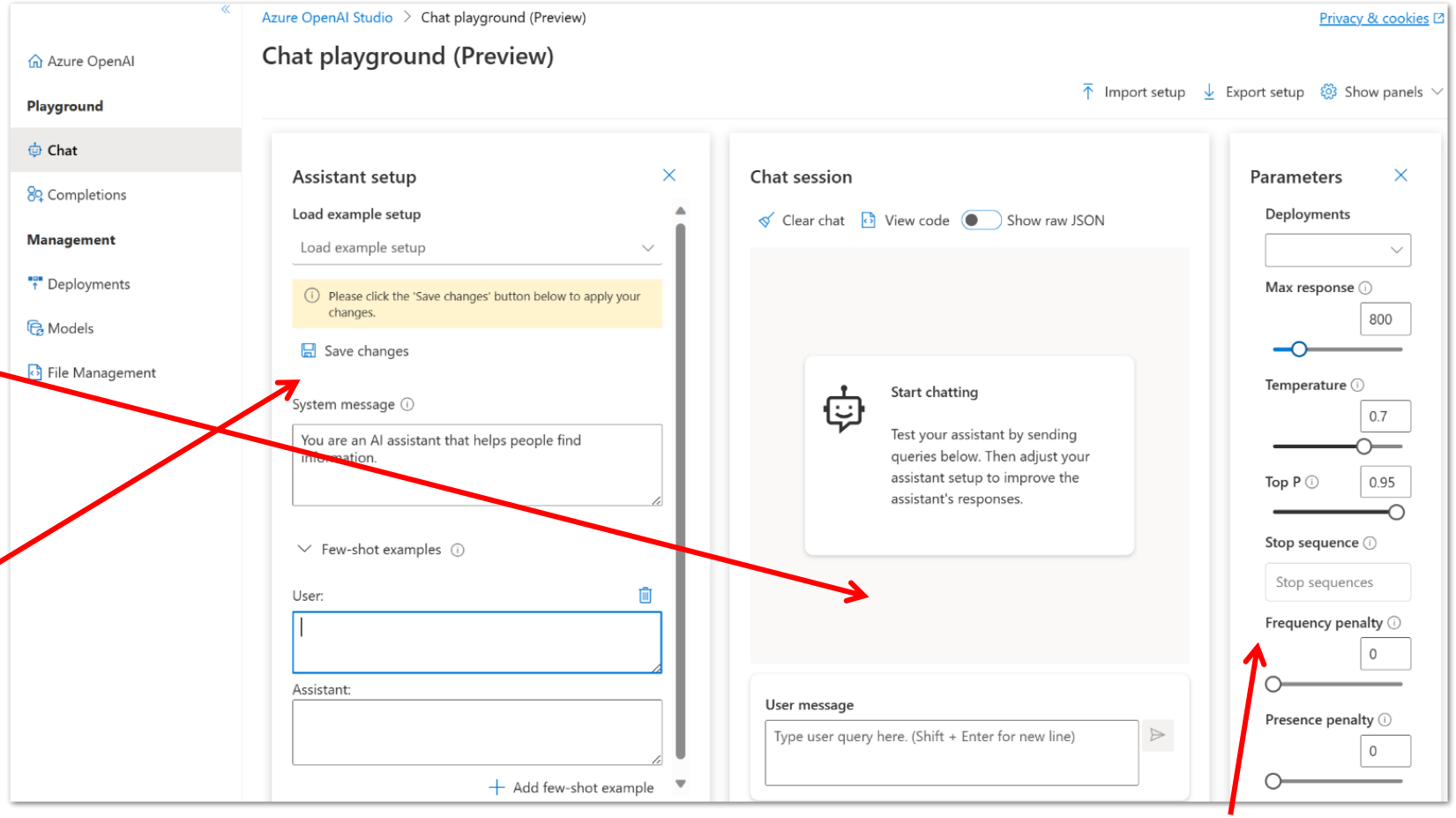
Azure OpenAI Studio

In **Azure AI Studio**, you can deploy **large language models**, provide few-shot examples, and **test** them in Azure OpenAI Studio's Chat playground.

The image shows **Azure OpenAI's Chat playground interface**, where users can test and configure an AI chatbot.

In the middle, there's a **chat area** to type user messages and see the assistant's replies.

On the left, there's a menu for **navigation** and a section to set up the **assistant**, including a reminder to save changes.



On the right, adjustable **parameters control** the AI's response behavior, like **length, randomness, and repetition**. Users input queries, adjust settings, and observe how the AI responds to fine-tune its performance.

Azure OpenAI Service pricing

Pricing for Language models

Models	Context	Prompt (Per 1,000 tokens)	Completion (Per 1,000 tokens)
GPT-3.5-Turbo	4K	$0.0015	$0.002
GPT-3.5-Turbo	16K	$0.003	$0.004
GPT-3.5-Turbo-1106	16K	N/A	N/A
GPT-4-Turbo	128K	N/A	N/A
GPT-4-Turbo-Vision	128K	N/A	N/A
GPT-4	8K	$0.03	$0.06
GPT-4	32K	$0.06	$0.12

Azure OpenAI Service pricing

Base models

Models	Usage per 1,000 tokens
Babbage-002	$0.0004
Davinci-002	$0.002

Fine-tuning models

Models	Training per compute hour	Hosting per hour	Input Usage per 1,000 tokens	Output Usage per 1,000 tokens
Babbage-002	$34	$1.70	$0.0004	$0.0004
Davinci-002	$68	$3	$0.002	$0.002
GPT-3.5-Turbo	$102	$7	$0.0015	$0.002

Image models

Models	Quality	Resolution	Price (per 100 images)
Dall-E-3	Standard	1024 * 1024	$4
	Standard	1024 * 1792, 1792 * 1024	$8
Dall-E-3	HD	1024 * 1024	$8
	HD	1024 * 1792, 1792 * 1024	N/A
Dall-E-2	Standard	1024 * 1024	$2

Embedding models

Models	Per 1,000 tokens
Ada	$0.0001

Speech models

Models	Per hour
Whisper	$0.36

What are copilots?

Copilots are a new type of computing tool that integrates with applications to help users with **common tasks using generative AI models**. They are designed using a standard architecture, allowing developers to create **custom copilots** tailored to specific business needs and applications.

- Copilots might appear as a chat feature beside your document or file, and they **utilize the content within the product to generate specific results**.

Creating a copilot involves several steps:

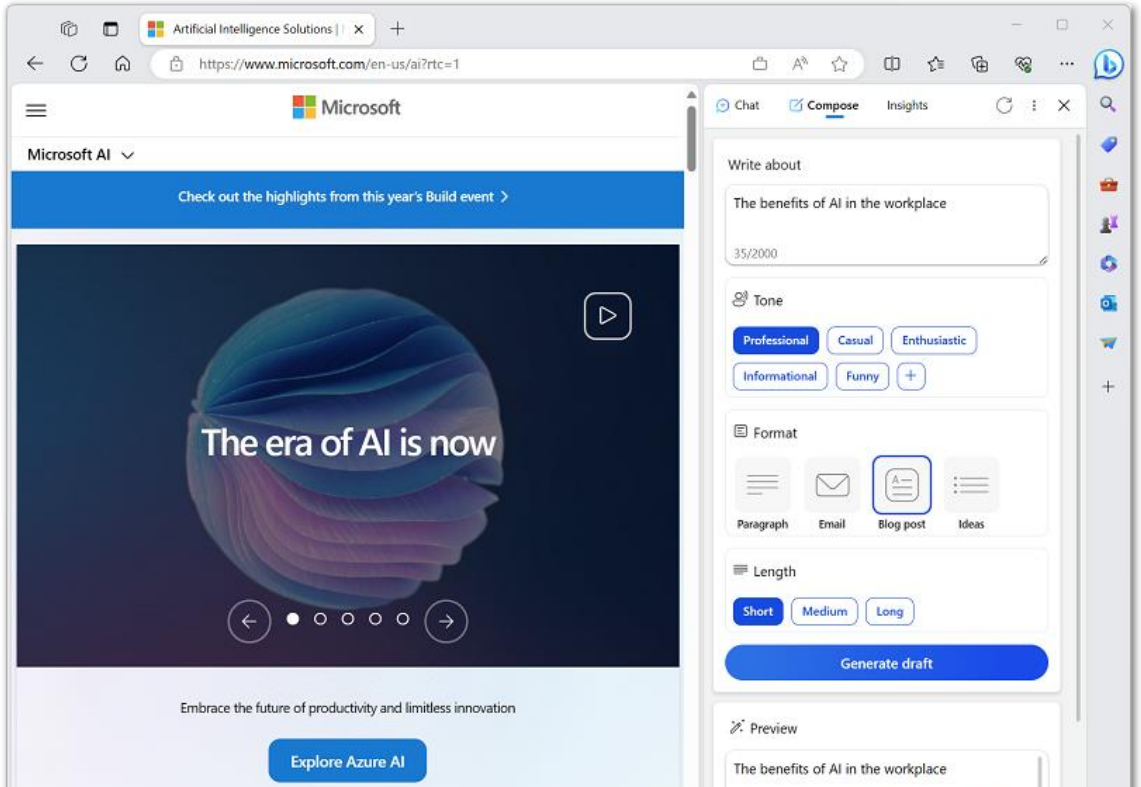
1. Training a **large language model** with a vast amount of data.
2. Utilizing services like **Azure OpenAI Service**, which provide pre-trained models that developers can either use as-is or fine-tune with their own data for more specific tasks.
3. **Deploying** the model to make it available for use within applications.
4. **Building copilots** that **prompt** the models to generate usable content.
5. Business users can use copilots to boost their **productivity and creativity** with AI-generated content.

Copilots have the potential to revolutionize the way we work. These copilots use generative AI to help with first drafts, information synthesis, strategic planning, and much more.

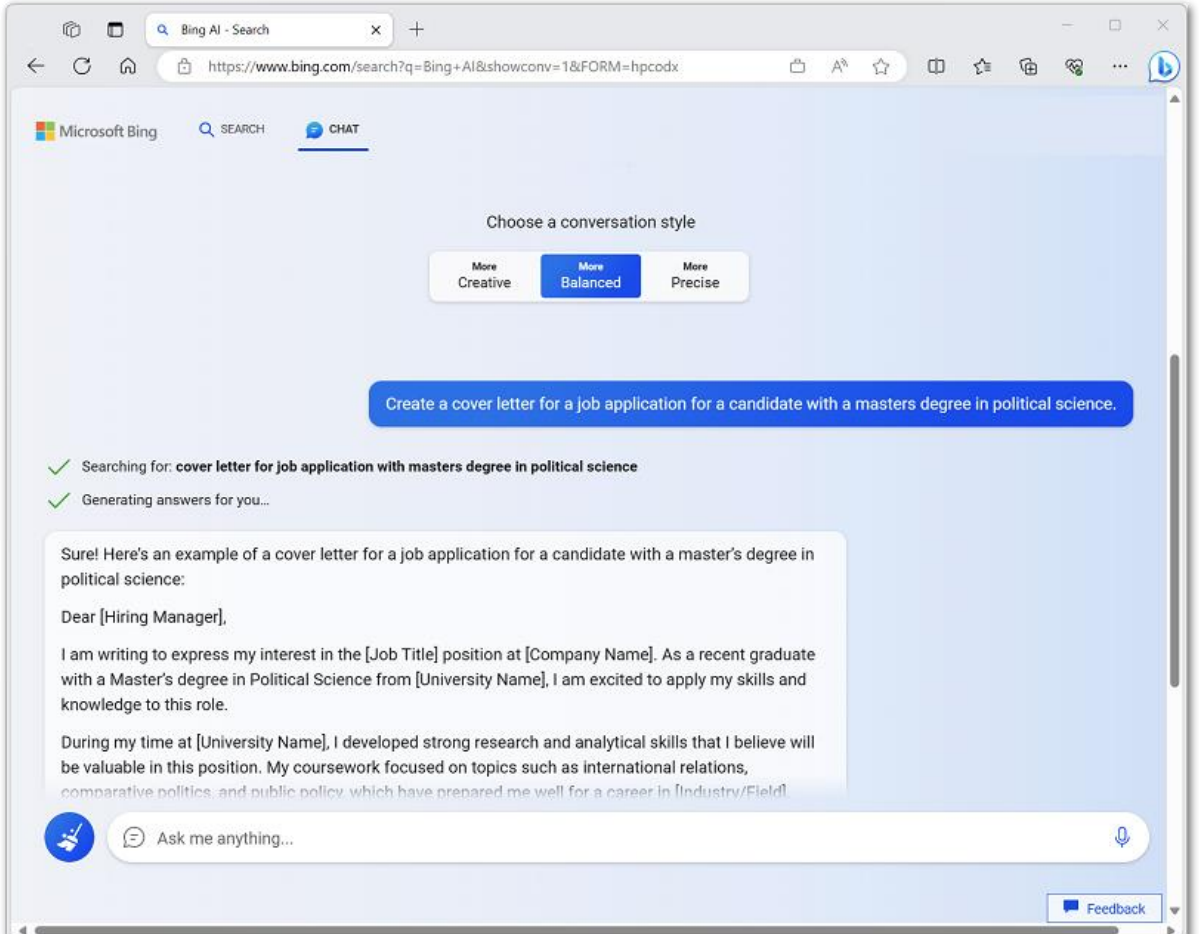
Copilot Examples

Microsoft Copilot is integrated into various applications to assist users in creating **documents, spreadsheets, presentations, and more**, by generating content, summarizing information, and aiding in strategic planning.

- It is used across Microsoft's suite of products and services to enhance user **experience and efficiency**.



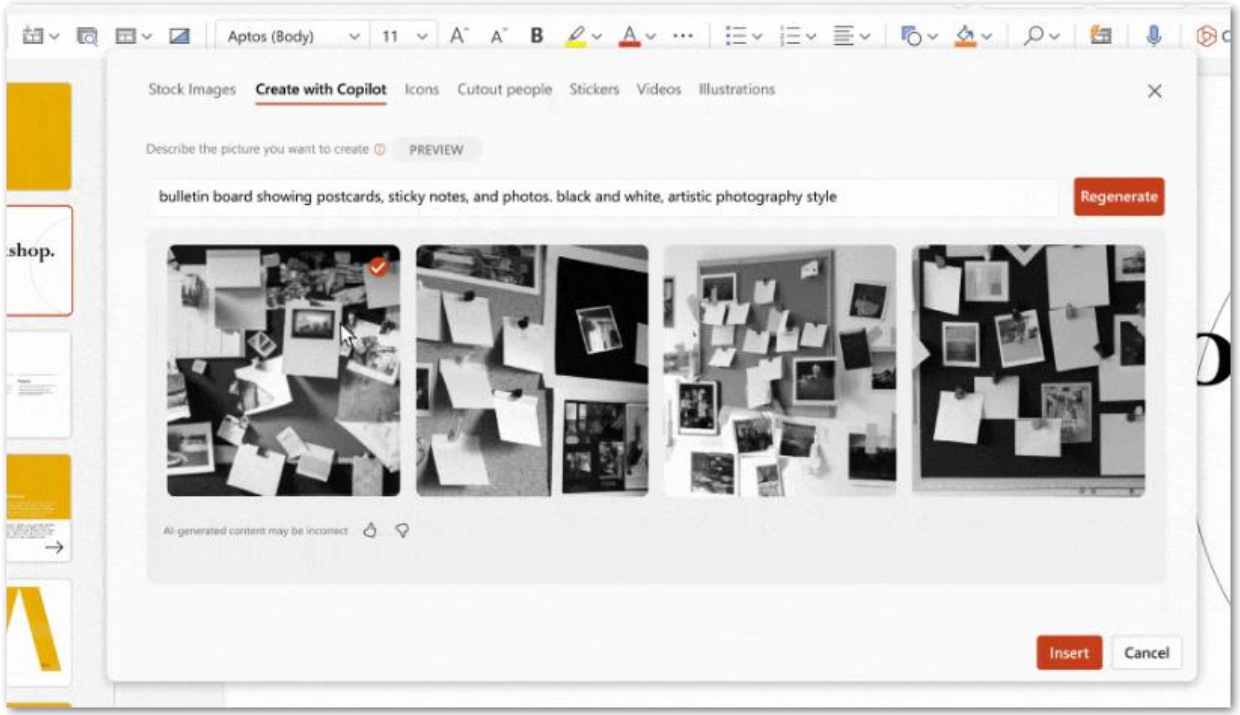
The Microsoft Bing search engine provides a copilot to help when **browsing or searching** the Internet by generating **natural language answers** to questions based on context rather than just search results of indexed pages.



Copilot Examples

Microsoft 365 Copilot is designed to be a partner in your workflow, integrated with productivity and communication tools like **PowerPoint** and **Outlook**.

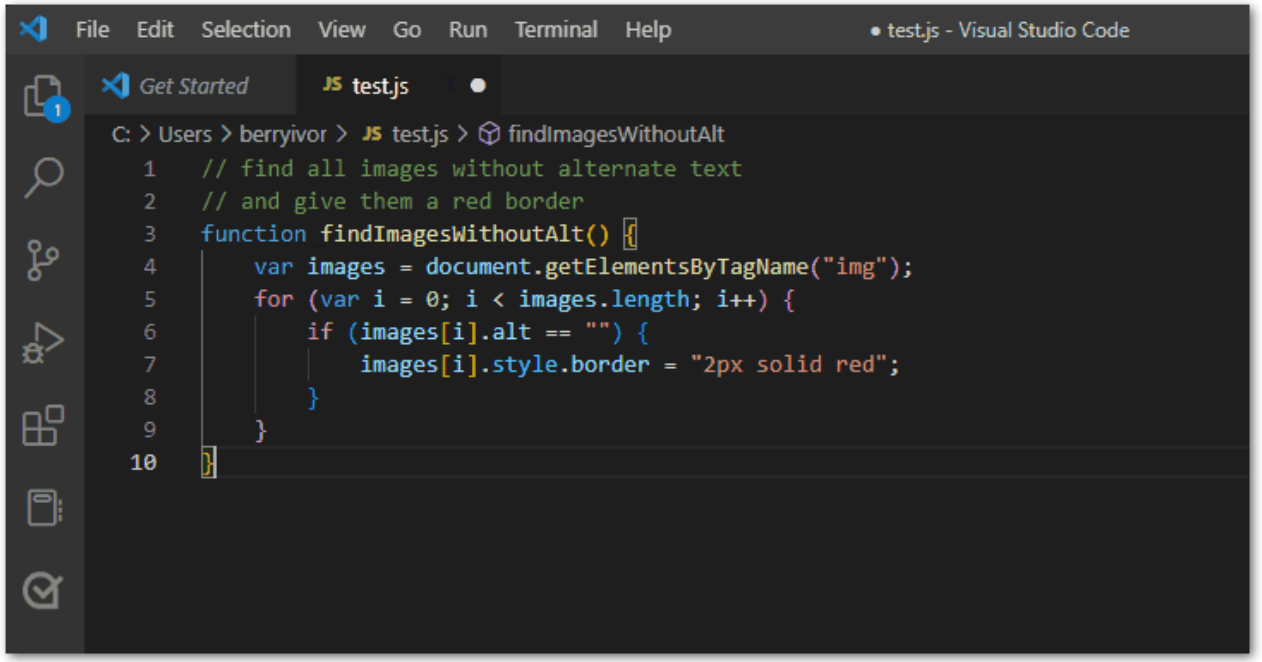
- It's there to help you craft effective **documents**, design **spreadsheets**, put together **presentations**, manage emails, and streamline other tasks.



GitHub Copilot

GitHub Copilot is a tool that helps software developers, offering real-time assistance as they **write code**. It offers more than suggesting code snippets; it can help in **documenting the code** for better understanding and maintenance.

- Copilot also helps **test out code**, which means coders can work smarter and make fewer mistakes.



Prompt engineering

Prompt engineering is a process that **improves the interaction between humans and generative AI**. It involves **refining the prompts** or instructions given to an AI application to **generate higher quality responses**. This process is valuable for both the developers who create AI-driven applications and the end-users who interact with them.

For example, developers may build a generative AI application for teachers to create multiple-choice questions related to text students read. During the development of the application, developers can **add other rules** for what the program should do with the prompts it receives.

System messages

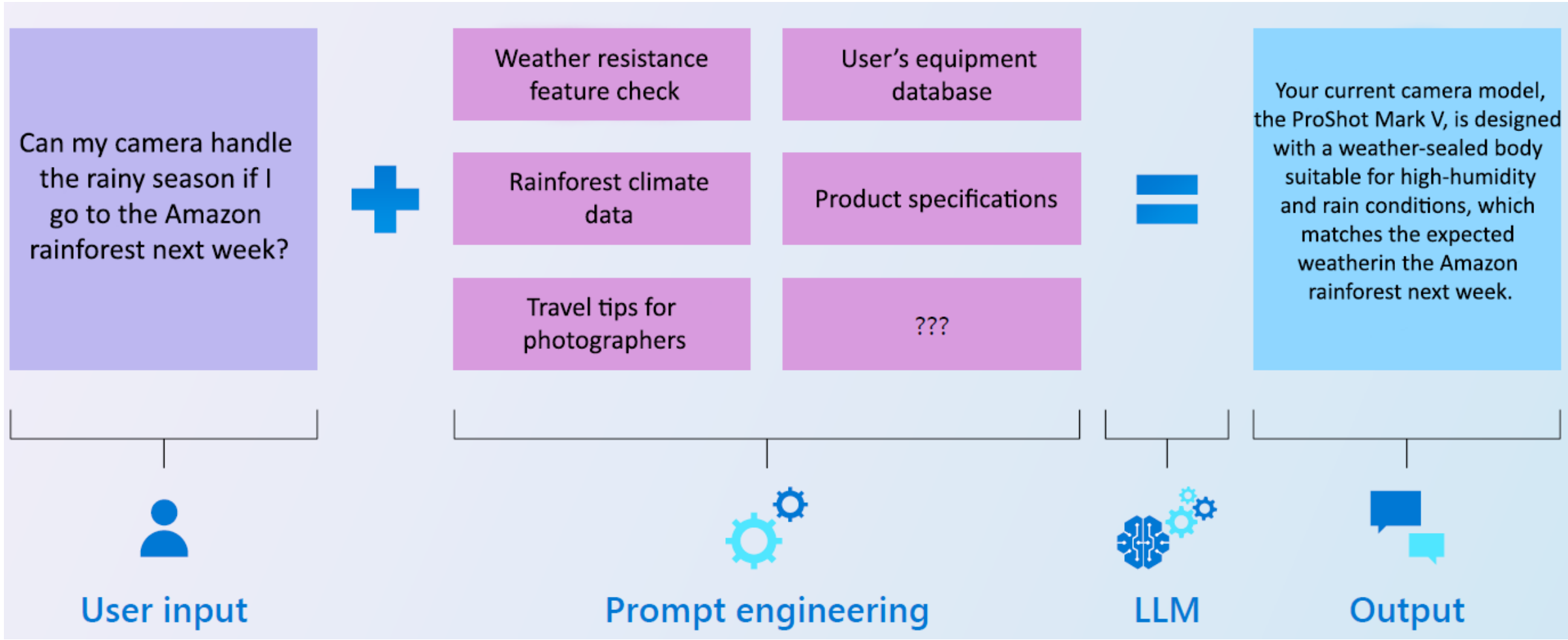
Prompt engineering techniques include defining a system message. The message sets the **context for the model** by describing **expectations** and **constraints**.
For example, "You're a helpful assistant that responds in a **cheerful, friendly manner**".
These system messages determine constraints and styles for the model's responses.

Writing good prompts

To maximize the utility of AI responses, it is essential to be **precise and explicit** in your prompts.
A well-structured prompt, such as **"Create a list of 10 things to do in Edinburgh during August,"** directs the AI to produce a targeted and relevant output, achieving better results.

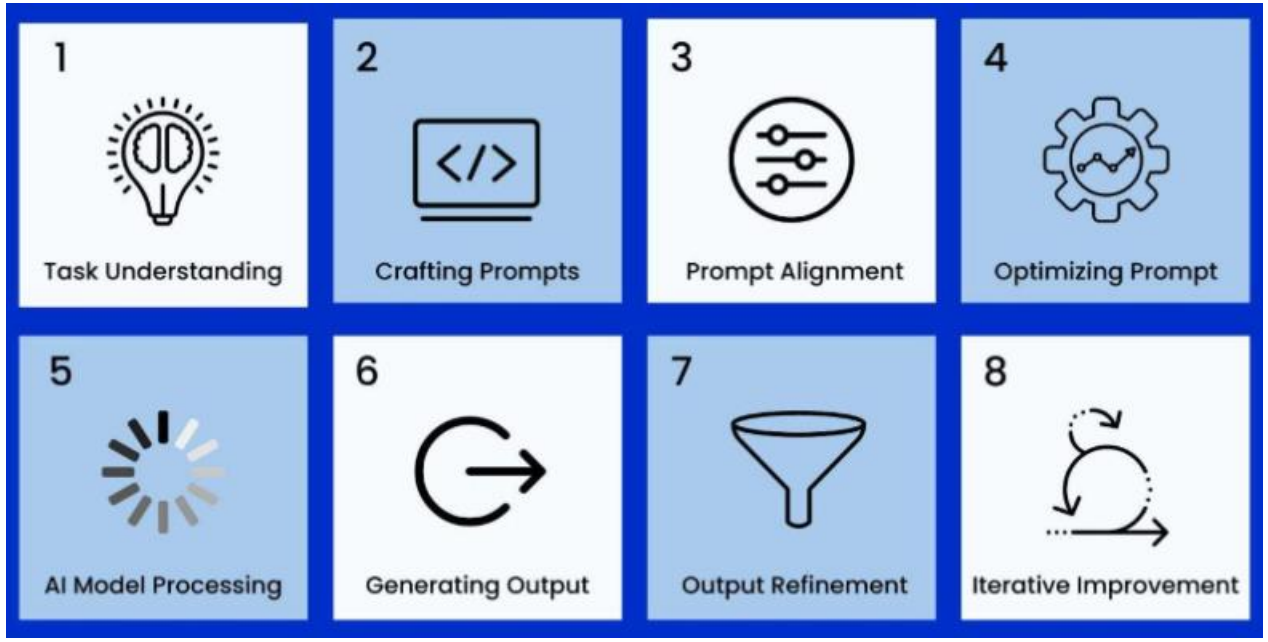
Prompt engineering

Zero-shot learning refers to an AI model's ability to correctly perform a task **without any prior examples or training** on that specific task.
One-shot learning involves the AI model learning from a **single example or instance** to perform a task.



Prompt engineering workflow

1. **Task Understanding**: Know what you want the AI to do.
2. **Crafting Prompts**: Write instructions for the AI.
3. **Prompt Alignment**: Make sure instructions match what the AI can do.
4. **Optimizing Prompt**: Improve the instructions for better AI responses.
5. **AI Model Processing**: The AI thinks about the instructions.
6. **Generating Output**: The AI gives an answer or result.
7. **Output Refinement**: Fix or tweak the AI's answer.
8. **Iterative Improvement**: Keep improving the instructions and answers.



Grounding

Grounding in prompt engineering is a technique used in large language models (LLMs) where you provide **specific, relevant context within a prompt**. This helps the AI to produce a more accurate and related response.

For example, if you want an LLM to summarize an email, you would include the actual email text in the prompt along with a command to summarize it. This approach allows you to leverage the LLM for tasks it wasn't explicitly trained on, without the need for retraining the model.

Prompt engineering vs Grounding

Prompt engineering broadly refers to the art of crafting effective prompts to produce the desired output from an AI model. **Grounding** specifically involves enriching prompts with relevant context to improve the model's understanding and responses.

Grounding ensures the AI has **enough information to process the prompt** correctly, whereas **prompt engineering** can also include techniques like format, style, and the strategic use of examples or questions to guide the AI.

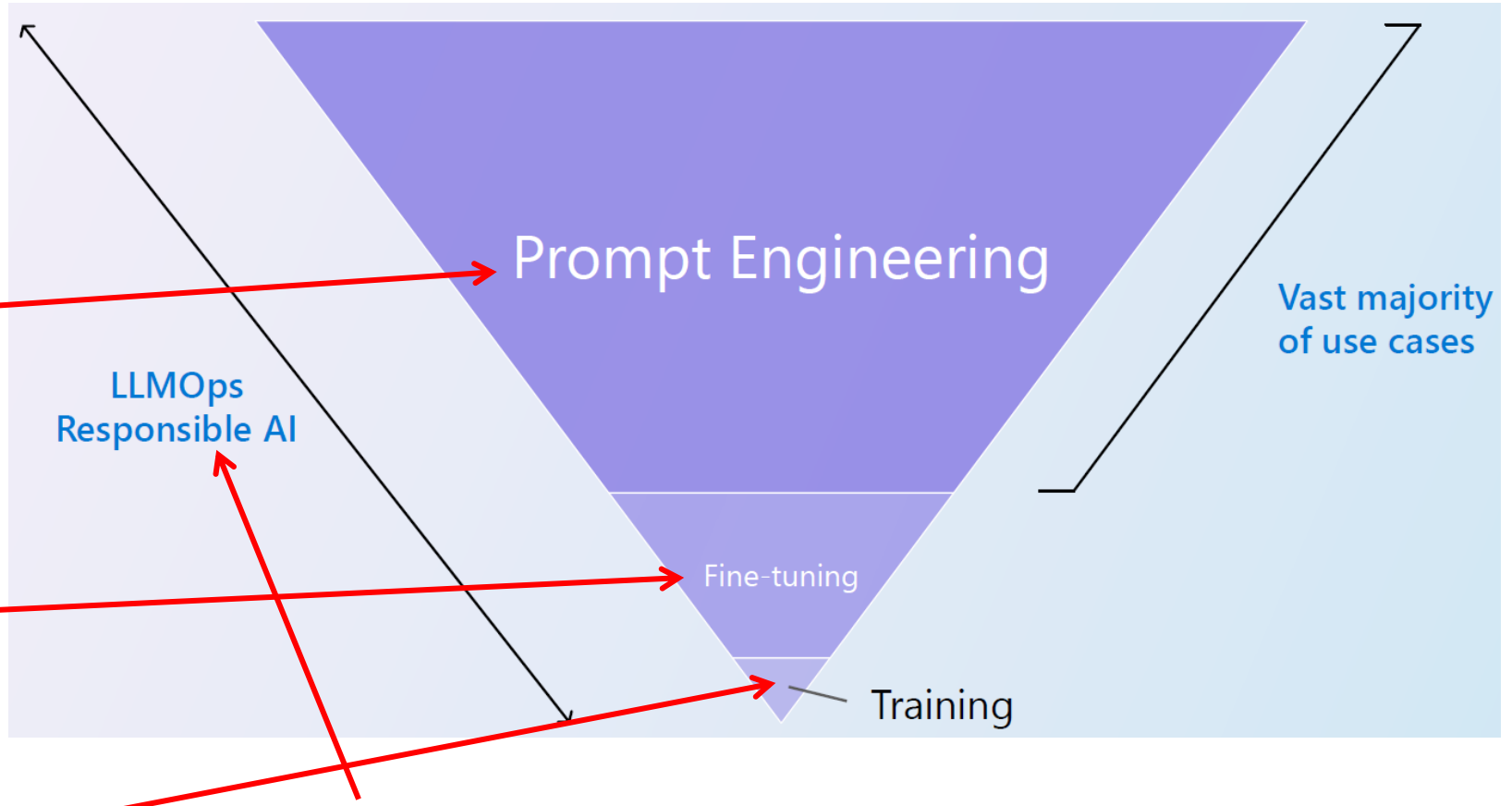
Grounding options

Grounding Options: These are techniques to ensure LLM outputs are accurate and adhere to responsible AI principles.

Prompt Engineering: Placed at the top, indicating its broad applicability, this involves designing prompts to direct the AI toward generating the desired output.

Fine-Tuning: A step below in complexity, where LLMs are trained on specific data to improve their task performance.

Training: The most resource-intensive process, at the triangle's base, suggesting its use in more extensive customization needs.



LLMOps and Responsible AI: These foundational aspects emphasize the importance of operational efficiency and ethical standards across all stages of LLM application development.

Responsible generative AI solution

The Microsoft guidance for responsible generative AI is designed to be practical and actionable. It defines a four stage process to develop and implement a plan for responsible AI when using generative models.
The four stages in the process are:

1. *Identify* potential harms that are relevant to your planned solution.
2. *Measure* the presence of these harms in the outputs generated by your solution.
3. *Mitigate* the harms at multiple layers in your solution to minimize their presence and impact, and ensure transparent communication about potential risks to users.
4. *Operate* the solution responsibly by defining and following a deployment and operational readiness plan.

Identify potential harms

The first stage in a responsible generative AI process is to identify the potential harms that could affect your planned solution. There are four steps in this stage, as shown here:

1. Identify potential harms
2. Prioritize identified harms
3. Test and verify the prioritized harms
4. Document and share the verified harms