

Connect to Azure OpenAI in Python

Created by David Song, May 25, 2025

Contents

Preface	1
Create a Resource on Azure Portal	1
Use Python to create a chat bot	4
Install Python packages.....	4
Create a file named: “.env”	4
Source code.....	5
Connect to Azure OpenAI	5
Required package.....	5
Source Code	5
Generate two dataframe differences report from OpenAI	6
Scouce code	6
result	7
References:	8

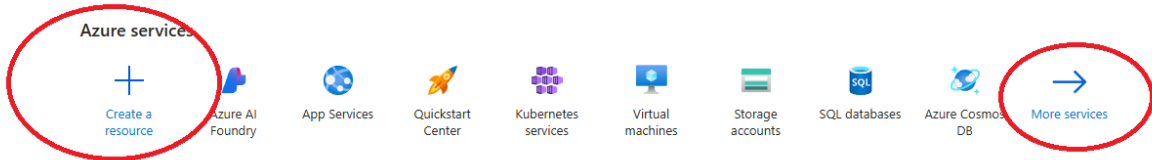
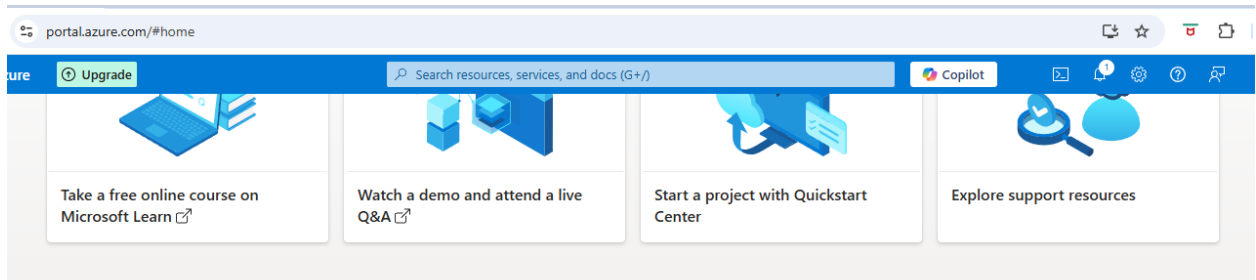
Preface

This document shows how to use Python to connect to Azure OpenAI. Generally speaking, you will need to create a resource in Azure, get the end point and api keys, and then use the end point and api leys to connect to Azure Open Ai. It’s simple like that.

There are some differences to use Azure OpenAI and use OpenAI directly. We will only focus on Azure OpenAI at this moment.

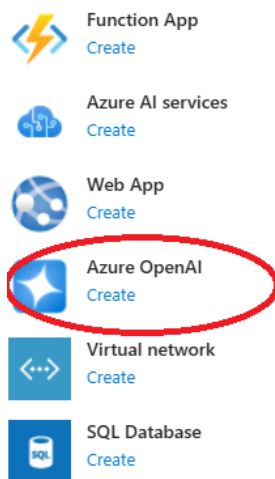
Create a Resource on Azure Portal

1. Login to Azure portal <https://portal.azure.com/#home> (Create a Azure account if you don’t have one).
2. Click on Create a resource or More Services:

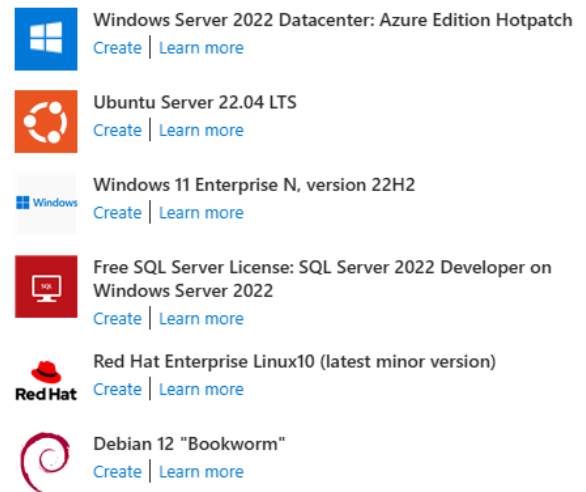


3. Click on Azure OpenAI

Popular Azure services [See more in All services](#)



Popular Marketplace products [See more in Marketplace](#)



4. Fill in information, then click next, next, ..., at the end click create button.

Create Azure OpenAI ...

1 Basics 2 Network 3 Tags 4 Review + submit

Azure OpenAI Service provides access to OpenAI's powerful language models, including all the latest OpenAI models. These models can be easily adapted to your specific tasks, including but not limited to content generation, summarization, image understanding, semantic search, and natural language to code translation. Top use cases include Call Centers, Virtual Assistants, Accessibility, Content Generation, and Code Development. The service also features the Assistants API, Fine Tuning capabilities and many ways to connect your data to the service for conversational experiences. The service can be scaled through Standard (tokens) and Provisioned (PTUs) deployment types.

[Learn more](#)

Project Details

Subscription * ⓘ

Azure subscription 1



Resource group * ⓘ

Hackenthon

[Create new](#)

Instance Details

Region ⓘ

East US

Name * ⓘ

DSAIPractice

Pricing tier * ⓘ

Standard S0

[View full pricing details](#)

Content review policy

5. Once the resource is created, it will be deployed. The deployment process will take a minute or two. Once it's done, click on resources, this will bring to the resource page. Click on the resource you just created.

You can also get to the page by clicking **Model + endpoints** at the left end corner of the resource page.

The screenshot shows the Azure AI Foundry interface for a deployment named 'gpt-4o'. The left sidebar contains navigation options like Home, Model catalog, Playgrounds, Build and customize, Agents, Templates, Fine-tuning, Observe and optimize, Monitoring, Protect and govern, Azure OpenAI Evaluation, Guardrails + controls, Risks + alerts, Governance, Azure OpenAI, Vector stores, and Data files. The main content area is divided into several sections:

- Endpoint:** Target URI is `https://dsong-mb199mtj-eastus2.cognitiveservices.azure.com/o...` and the Key is masked with dots.
- Deployment info:** A table showing details:

Name gpt-4o	Provisioning state Succeeded
Deployment type Global Standard	Created on 2025-05-23T20:32:57.9261614Z
Created by dsong99@gmail.com	Modified on May 23, 2025 4:32 PM
Modified by dsong99@gmail.com	Version upgrade policy Once a new default version is available
Rate limit (Tokens per minute) 50,000	Rate limit (Requests per minute) 300
Model name gpt-4o	Model version 2024-11-20
Life cycle status Generally Available	Date created Dec 2, 2024 7:00 PM
Date updated Dec 2, 2024 7:00 PM	Model retirement date Feb 28, 2026 7:00 PM
- Monitoring & safety:** Content filter is set to DefaultV2.
- Get Started:** Includes sections for authentication using API key and a basic code sample.
 - 1. Authentication using API Key:** Explains how to use the endpoint URL and API key for authentication.
 - 2. Install dependencies:** Shows the command `pip install openai`.
 - 3. Run a basic code sample:** Provides a Python script snippet for making a synchronous call to the chat completion API.

- On top of the page, you can see Endpoint and key, these two are needed in your python script, just copy those two values. On the right side of the page, it shows examples how to connect to Azure OpenAI.

Use Python to create a chat bot

Install Python packages

This example use a few Python packages, you will need to install them either use conda or pip: dotenv, langchain, langchain_openai

Create a file named: ".env"

Copy and paste the endpoint and key from resource page above to this file:

```
AZURE_OPENAI_API_KEY='AxYnHXDtx***'
AZURE_OPENAI_ENDPOINT='https://dsong-***version=2025-01-01-preview'
```

Source code

```
from langchain_openai import AzureChatOpenAI
from dotenv import load_dotenv
from langchain.prompts import ChatPromptTemplate

load_dotenv()

llm = AzureChatOpenAI(
    azure_deployment="gpt-4o", # or your deployment
    api_version="2025-01-01-preview", # or your api version
)

user_input=input('You:')

prompt = ChatPromptTemplate(
    {
        ('system', ('You are a helpful AI assistant. Please assist the
user')),
        ('human', user_input)
    }
)

response = llm.invoke(prompt.format(user_input=user_input))
print(response.content)
```

1. Note: load_dotenv() loads the .env file and set properties in that file to the environment.

Connect to Azure OpenAI

This example will show how to connect to AzureOpenAI without using langchain.

Required package: install openai

Source Code

```
import os
from openai import AzureOpenAI

endpoint = 'https://dsong-mb199mtj-*** 2025-01-01-preview'
model_name = "gpt-4o"
deployment = "gpt-4o"

subscription_key = 'AxYnHXDtxFCi1uuKKk3r7rk***'
api_version = "2025-01-01-preview"

client = AzureOpenAI(
    api_version=api_version,
```

```

        azure_endpoint=endpoint,
        api_key=subscription_key,
    )

    response = client.chat.completions.create(
        messages=[
            {
                "role": "system",
                "content": "You are a helpful assistant.",
            },
            {
                "role": "user",
                "content": "I am going to Paris, what should I see?",
            }
        ],
        max_tokens=10,
        temperature=0,
        top_p=1.0,
        model=deployment
    )

    print(response.choices[0].message.content)

```

Generate two dataframe differences report from OpenAI

Scouce code

```

import os
from openai import AzureOpenAI

endpoint = 'https://dsong-***-eastus2.cognitiveservices.azure.com/openai/deployments/gpt-4o/chat/completions?api-version=2025-01-01-preview'
model_name = "gpt-4o"
deployment = "gpt-4o"

subscription_key = 'AxYnHxD***'
api_version = "2025-01-01-preview"

client = AzureOpenAI(
    api_version=api_version,
    azure_endpoint=endpoint,
    api_key=subscription_key,
)
import pandas as pd

df1 = pd.DataFrame({'A': [1, 2, 3], 'B': [4, 5, 6]})
df2 = pd.DataFrame({'A': [1, 2, 5], 'B': [4, 7, 6]})

comparison = df1.compare(df2)
prompt = f"""

```

Compare the following two dataframes and generate a report highlighting the differences:

```
DataFrame 1:  
{df1.to_string() }
```

```
DataFrame 2:  
{df2.to_string() }
```

Generate a concise report summarizing the changes, including added, removed, and modified rows or values.
"""

```
response = client.chat.completions.create(  
    messages=[  
        {  
            "role": "system",  
            "content": "You are a helpful assistant.",  
        },  
        {  
            "role": "user",  
            "content": prompt,  
        }  
    ],  
    max_tokens=200,  
    temperature=0,  
    top_p=1.0,  
    model=deployment  
)
```

```
# Extract the generated report  
print(response.choices[0].message.content)
```

result

Report: Comparison of DataFrame 1 and DataFrame 2

Summary of Differences:

1. **Modified Values**:

- In row index `1`, column `B`:
 - **DataFrame 1**: Value is `5`
 - **DataFrame 2**: Value is `7`

2. **Modified Rows**:

- Row index `2`:
- `**DataFrame 1**`: `[A=3, B=6]`
- `**DataFrame 2**`: `[A=5, B=6]`

3. `**No Added or Removed Rows**`:

- Both DataFrames have the same number of rows (3). No rows were added or removed.

Detailed Changes:

Index	Column	DataFrame 1 Value	DataFrame 2 Value	Change Type
-----	-----	-----	-----	-----
1	B	5		

References:

1. OpenAI document <https://platform.openai.com/docs/overview>
2. Python connect to OpenAI: <https://www.youtube.com/watch?v=czvVibB2IRA>
3. Tutorial video on Azure OpenAI and GPT Models <https://www.youtube.com/watch?v=jQyYeYWD97I>
4. Tutorial video Getting Started with Azure OpenAI | GPT 4o
https://www.youtube.com/watch?v=H_1Ge6wxaaE
5. How to setup Azure OpenAI service and use it in Python
<https://www.youtube.com/watch?v=50ZwmkCvE88>
6. Free AI: <https://huggingface.co/>
7. Python langchain package:
https://python.langchain.com/docs/integrations/chat/azure_chat_openai/