How To Spin Up Dev Environments

Table of Contents

Introduction

This documentation serves as a practical guide for setting up resources in your MLOps (Machine Learning Operations) environment. Our goal is to facilitate the efficient and structured provisioning of resources to support the development, quality assurance (QA), and production stages of your machine learning projects.

In this guide, we provide clear, step-by-step instructions and best practices to ensure your MLOps environment is tailored to your specific needs. Whether you're managing data, training models, or deploying solutions, this documentation will help you streamline the resource setup process and deliver exceptional results.

What to Expect

Within this guide, you will find practical guidance on setting up your MLOps environments:

1. **Prerequisites and Infrastructure**: Get started by understanding the prerequisites and requirements for setting up your MLOps resources. Ensure you have the foundational elements in place.

2. **Environment Configuration**: Learn how to configure the initial environment. This includes setting up cloud infrastructure and creating the necessary virtual environments.

3. **Resource Provisioning**: Walk through the process of provisioning resources, from cloud services to software installations, ensuring you have the essential components ready.

4. **Environment Differentiation**: Explore strategies for differentiating your Dev, QA, and Prod environments to match the specific needs of each stage.

6. **Scalability and Resource Management**: Discover how to scale your resources and manage them efficiently to meet changing demands.

7. **Monitoring and Compliance**: Set up monitoring and compliance measures to track performance, detect anomalies, and adhere to regulations.

8. **Automation and CI/CD**: Implement automation and CI/CD pipelines for efficient resource management and deployments.

9. **Documentation and Best Practices**: Embrace the best practices for documentation and knowledge sharing to maintain a well-organized and transparent environment.

Machine Learning Models

There are three types of ML models concerned for handling.

1. Statistical Models
    a. Examples: Forecasting models like ARIMA, ETS and different regression models
2. Azure AI Cognitive Search Services based Models
    a. Examples: ChatBOT models on Custom Databases.

3. Deep learning models (from scratch)
    a. Examples: Convolutional models like CNN, LSTM, BERT, Generative models like GAN.

Semi-Supervised Learning and Reinforcement Learning are other machine learning paradigms that combine aspects of both supervised and unsupervised learning. Semi-supervised learning leverages a mix of labeled and unlabeled data, while reinforcement learning focuses on training agents to make sequential decisions through trial and error.

Statistical Models

Statistical models are a crucial component of data-driven decision-making processes. They enable organizations to predict future trends, classifying behaviors, or events based on historical data and patterns. These models are used to anticipate changes in various domains, including finance, sales, inventory management, weather forecasting, and more.

Based on training input data, Machine Learning models are classified into Supervised, Unsupervised and Semi-supervised models.

Supervised Models

1. **Definition**: Supervised models follow a machine learning paradigm where the algorithm is trained on a labeled dataset, meaning the input data is paired with corresponding output labels. The goal is to learn a mapping from inputs to outputs, allowing the algorithm to make predictions or classifications for new, unlabeled data.
2. **Key Characteristics**:
    a. Labeled Data: Training data consists of input-output pairs, where the output (target or label) is known.
    b. Objective: The objective is to learn a model that can accurately predict the output for new, unseen data.
    c. Types: Common supervised learning tasks include classification (predicting categories) and regression (predicting numerical values).
3. **Examples of Supervised Learning**:
    a. Email Spam Detection: Given labeled examples of spam and non-spam emails, a supervised learning algorithm can learn to classify incoming emails.
    b. Image Classification: Supervised learning can be used to classify images into predefined categories, such as identifying animals in photos.
    c. Predicting House Prices: Using historical data with features like size, location, and number of bedrooms, supervised regression can predict house prices.

Classification Model

- Classification models are a subset of supervised machine learning algorithms that are used when the target variable represents categories or classes. These models aim to assign input data points to one of several predefined classes. For instance, they can be employed to classify emails as spam or not, identify the species of a plant based on its features, or predict whether a customer will churn or stay with a service. Sample examples of classification algorithms include logistic regression, decision trees, k-nearest neighbors, and deep learning models like convolutional neural networks (CNNs) for image classification. In a statistical context, classification is akin to discriminant analysis, where the goal is to differentiate observations into groups based on their characteristics.

Following are some examples for Classification based models.

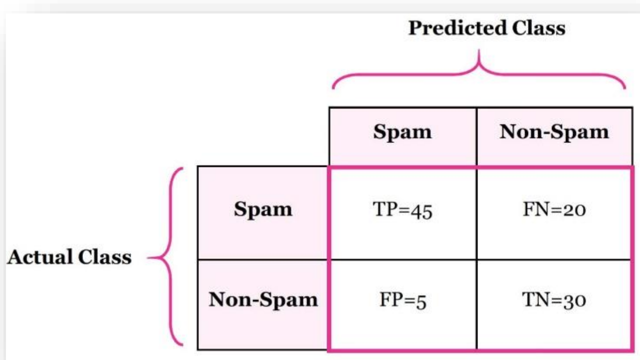
1. Support Vector Machine (SVM) models
    a. Support Vector Machines are widely popular statistical models for pattern recognition in data. Simple and logistic regression are not able to model the data for higher dimensions.
    b. Support Vector Machine algorithms can model the data having multiple complexities which is not a linear model.
    c. These models were used widely before the Deep Learning models. Use cases of these models comprises A-Z like Spam mail detection, text classification, image classification etc.
    d. These models can be used for Image segmentation. To separate foreground data with background data.
2. Logistic regression models are majorly classification-based model. Output generated by it is majorly a class or multiple classes.
    i. If we have details for flower color, flow size (height, width, length) and flower class, then we can train one logistic regression model on the data.
    ii. It takes input different features like flower size, color etc. and it provides flower class like Rose, Lily in the output.
3. Apart from these following are also few very high performing classification models.
    a. Random Forest
    b. Naive Bayes
    c. Gradient Boosting (e.g., XGBoost, LightGBM)
    d. K-Nearest Neighbors (K-NN)
    e. Decision Trees
    f. LDA (Linear Discriminant Analysis)
    g. QDA (Quadratic Discriminant Analysis)
    h. Gaussian Process Classification

Commonly used measurement metrics for classification models are as follows:

- **Accuracy**
    a. Accuracy measures the proportion of correct predictions out of the total predictions.
    b. It is a straightforward metric, but it may not be suitable for imbalanced datasets, where one class dominates.
- **Precision**
    a. Precision, also known as positive predictive value, measures the accuracy of positive predictions made by the model.
    b. It is the ratio of true positives to the sum of true positives and false positives.
    c. High precision indicates that the model makes few false positive errors.
- **Recall (Sensitivity)**

a. Recall, also known as true positive rate or sensitivity, measures the ability of the model to identify all relevant instances in the dataset.
b. It is the ratio of true positives to the sum of true positives and false negatives.
c. High recall indicates that the model captures a high proportion of actual positive instances.

- **F1-Score**
  a. The F1-Score is the harmonic mean of precision and recall and provides a balance between the two metrics.
  b. It is useful when precision and recall need to be balanced, and there is a trade-off between them.
- **Specificity (True Negative Rate)**
  a. Specificity measures the ability of the model to correctly identify negative instances.
  b. It is the ratio of true negatives to the sum of true negatives and false positives.
- **ROC Curve (Receiver Operating Characteristic)**
  a. The ROC curve is a graphical representation of the trade-off between true positive rate (recall) and false positive rate at different thresholds.
  b. The Area Under the Curve (AUC) is often used to summarize the overall performance of the model.
- **Confusion Matrix**
  a. A confusion matrix provides a tabular summary of the model's performance, showing the counts of true positives, true negatives, false positives, and false negatives.



- **Cohen's Kappa**
  a. Cohen's Kappa measures the agreement between the model's predictions and the actual outcomes while accounting for the possibility of agreement occurring by chance.
- **Matthews Correlation Coefficient (MCC)**
  a. MCC is a correlation coefficient that measures the quality of binary classifications, taking into account all four elements of the confusion matrix.
- **Log-Loss (Cross-Entropy Loss)**

   a. Log-loss quantifies the accuracy of the model's predicted probabilities compared to the true probabilities.
   b. It is commonly used in probabilistic classification problems.
- **Precision-Recall Curve**
   a. The precision-recall curve illustrates the trade-off between precision and recall at different probability thresholds.

More details can be checked in the Microsoft Documentation.

Regression Models

- Regression models are another category of supervised machine learning algorithms that are applied when the target variable represents continuous, numeric values. These models aim to predict a value within a range. Regression is used in various scenarios, such as forecasting stock prices, estimating the price of a house based on its characteristics, or predicting the temperature. Sample examples of regression algorithms include linear regression, decision tree regression, support vector regression, and deep learning models like recurrent neural networks (RNNs) for time series prediction. In statistical terms, regression encompasses linear regression, where the focus is on modeling the relationship between a dependent variable and one or more independent variables to make predictions or infer associations.

Following are some examples for regression models.

4. Simple and logistic regression models
   a. These models are supervised models and requires output labels while training each input.
   b. Simple Regression models
      i. These are generalized model which tries to fit a line or curve among the data points.
      ii. For example, simple regression model for house price prediction, takes room size, room numbers, square ft. area of house, locality of the house and other details. It trains over many such houses details and provides an approximation for price rent if we provide it a required house size, room number and all other parameters.
      iii. Output generated by these models is majorly numerical data like price of house, approximate weight of a baby etc.
   c. Apart from these, following are other high performing regression models.
      i. Ridge Regression
      ii. Lasso Regression
      iii. Elastic Net
      iv. Decision Trees
      v. Random Forest
      vi. Gradient Boosting (e.g., XGBoost, LightGBM)
      vii. Support Vector Regression (SVR)
      viii. Principal Component Regression (PCR)
      ix. Partial Least Squares (PLS)

x. Gaussian Process Regression
xi. Quantile Regression

Measurement metrics for regression models

- **Mean Absolute Error (MAE)**
  a. MAE calculates the average absolute difference between the predicted values and the actual values.
  b. It provides a measure of the model's accuracy in terms of the actual target scale.
  c. MAE is less sensitive to outliers compared to some other metrics.
- **Mean Squared Error (MSE)**
  a. MSE calculates the average of the squared differences between the predicted values and the actual values.
  b. It gives more weight to larger errors, making it sensitive to outliers.
  c. RMSE (Root Mean Squared Error) is the square root of MSE and provides an error measure in the same units as the target variable.
- **R-squared (R^2)**
  a. R-squared measures the proportion of the variance in the target variable that is explained by the model.
  b. It ranges from 0 to 1, with higher values indicating a better fit.
  c. R-squared is often used to assess how well the model captures the variation in the data.
- **Adjusted R-squared**
  a. Adjusted R-squared is a modified version of R-squared that takes into account the number of predictor variables in the model.
  b. It penalizes models that include unnecessary features and provides a better measure of model fit in multiple regression.
- **Mean Absolute Percentage Error (MAPE)**
  a. MAPE calculates the average percentage difference between the predicted and actual values.
  b. It provides a measure of the model's accuracy in percentage terms, which can be useful when interpreting errors in real-world scenarios.
- **Root Mean Squared Logarithmic Error (RMSLE)**
  a. RMSLE calculates the RMSE of the natural logarithm of the predicted and actual values.
  b. It is commonly used when the target variable follows a skewed distribution and can help mitigate the impact of extreme values.
- **Mean Bias Deviation (MBD)**
  a. MBD measures the average difference between the predicted and actual values.
  b. It indicates whether the model tends to underpredict or overpredict.
- **Coefficient of Determination (COD)**
  a. COD is another measure of explained variance and is similar to R-squared but not bounded between 0 and 1.
  b. It can provide insights into the strength and direction of the relationship between predictors and the target.

- **Akaike Information Criterion (AIC) and Bayesian Information Criterion (BIC)**
    a. AIC and BIC are used for model selection and comparison in regression.
    b. Lower values of AIC and BIC indicate a better fit while penalizing models with excessive complexity.
- **Residual Analysis**
    a. Analyzing residuals, which are the differences between predicted and actual values, is a fundamental way to assess the quality of a regression model. Residual plots, such as a histogram or Q-Q plot, can reveal patterns or anomalies in the model's performance.

More details of measurement metrics can be found at Microsoft Documentation.

Time Series Forecasting Models

- Time series forecasting models are a subset of regression models that focus on predicting future values of a variable based on its historical data, typically ordered by time. Unlike traditional regression, these models consider temporal dependencies and trends, making them suitable for applications like stock price predictions, weather forecasting, and demand forecasting. Time series regression models leverage techniques such as autoregressive (AR), moving average (MA), and autoregressive integrated moving average (ARIMA) to capture and forecast patterns and fluctuations over time, making them a crucial tool for making predictions in dynamic, time-ordered datasets.

At their core, forecasting models are designed to answer questions like:

- "What will our sales look like in the next quarter?"
- "How will demand for a product change in the coming months?"
- "What will the weather be like next week?"
- "How will stock prices fluctuate in the future?"
- "Is there a statistically significant relationship between a student's study hours and their exam scores? "
- "The plant is 6 ft tall with green color and stick type structure, what could be its name? "

These models are essential for making informed decisions, optimizing resource allocation, and proactively addressing challenges in numerous industries.

Model Characteristics

1. Static code created using Python or R, which runs at a specific time frequency (daily/monthly/yearly).
2. Input is a csv file, or a tabular data having one or more features.
3. Output is future month forecasting, majorly a csv or tabular format.

Following are some examples for forecasting models.

1. Forecasting models like Autoregressive Integrated Moving Average (ARIMA), Exponential Smoothing (ETS), Seasonal AutoRegressive Integrated Moving Average (SARIMA)

    a. ETS, ARIMA/SARIMA models are regression based model used for forecasting. It takes Date, Revenue data along with some other columns/features and generates the forecast for next 6 to 12 months. It considers trend, seasonal properties of data for forecasting.
2. Apart from these, following are few very high performing forecasting models.
    a. Prophet
    b. State Space Models
    c. GARCH (Generalized Autoregressive Conditional Heteroskedasticity)
    d. Holt-Winters Exponential Smoothing
    e. Vector Autoregression (VAR)
    f. SARIMA (Seasonal ARIMA)
    g. BATS (Box-Cox Transformations, ARMA Errors, Trend, and Seasonal Components)
    h. TBATS (Trigonometric Seasonal Components, Box-Cox Transformations, ARMA Errors, Trend, and Seasonal Components)

Please refer to the references section for the details for mentioned algorithms.

This documentation provides details about evaluation metrics for time series forecasting models.

Apart from these categories, there is feedback learning methods which are part of Supervised learning.

Reinforcement Learning

Reinforcement learning is a machine learning paradigm where an agent learns to make a sequence of decisions by interacting with an environment. The agent receives feedback in the form of rewards or punishments based on its actions, and its goal is to learn a policy that maximizes the cumulative reward over time. Unlike supervised learning, reinforcement learning doesn't rely on labeled data but instead focuses on exploration and trial-and-error learning. This approach is commonly used in solving problems where an agent needs to learn how to make a series of decisions to achieve specific goals, such as game playing, robotic control, and autonomous decision-making

Reinforcement learning has seen the development of various algorithms designed to tackle different aspects of the learning process. Some notable reinforcement learning algorithms include:

- **Q-Learning**
    - Q-learning is a model-free algorithm used for discrete state and action spaces. It learns a Q-value function to estimate the expected future rewards for each state-action pair and is widely used in many applications.
- **Deep Q-Networks (DQN)**
    - DQN is an extension of Q-learning that uses deep neural networks to approximate the Q-value function, making it suitable for environments with high-dimensional state spaces, such as video games.
- **Policy Gradient Methods**
    - These include algorithms like REINFORCE and TRPO (Trust Region Policy Optimization), which directly learn the policy to maximize expected rewards.
- **Actor-Critic Methods**

    - Actor-Critic algorithms combine the advantages of both policy-based and value-based methods. They consist of an actor (policy) and a critic (value function) that work in tandem to improve the policy.
- **Proximal Policy Optimization (PPO)**
    - PPO is a popular policy optimization algorithm that balances ease of implementation and sample efficiency. It often delivers good performance and stability.
- **A3C (Asynchronous Advantage Actor-Critic)**
    - A3C is an asynchronous variant of the actor-critic method designed for parallel and distributed reinforcement learning, making it more efficient in large-scale environments.
- **Dueling DQN**
    - Dueling DQN is an extension of DQN that separates the value and advantage functions, improving the stability and convergence speed of the learning process.
- **Soft Actor-Critic (SAC)**
    - SAC is an off-policy actor-critic algorithm that addresses exploration challenges and is effective in continuous action spaces with high-dimensional inputs.
- **Twin Delayed DDPG (TD3)**
    - TD3 is an off-policy actor-critic method designed to enhance stability and convergence in continuous action environments.
- **Monte Carlo Methods**
    - Monte Carlo methods estimate the expected return from sampled trajectories and are often used in situations where the environment dynamics are not completely known.

Unsupervised Models

1. Unsupervised learning is a machine learning approach where the algorithm is provided with unlabeled data, and its objective is to discover patterns, structures, or groupings within the data without explicit guidance.
2. Key Characteristics:
    a. Unlabeled Data: Unlike supervised learning, the input data lacks associated output labels.
    b. Objective: The primary goal is to uncover hidden structures, clusters, or relationships in the data.
    c. Types: Unsupervised learning tasks include clustering (grouping similar data points) and dimensionality reduction (simplifying data while preserving important information).
3. Examples of Unsupervised Learning:
    a. Customer Segmentation: Unsupervised clustering can group customers based on their purchasing behaviors, helping businesses tailor marketing strategies.
    b. Topic Modeling: Unsupervised learning can be used to discover common topics within a collection of documents, making it useful for document categorization.
    c. Anomaly Detection: Detecting rare and unusual patterns in data without labeled anomalies, such as identifying fraudulent transactions in financial data.

Following are examples of unsupervised models.

Dimensionality Reduction Models

1. Principal Component Analysis (PCA) Models
    a. These models are used for Dimensionality reduction purpose.
    b. When we want classify different people face from large sized images, we can use PCA to reduce the dimensionality and highlight the required features in the data.
    c. These models take vectors like images in the input and provides smaller dimension vector or matrix. If an image is having 1000x1000 dimension, PCA output will have lesser, can be a dimension like 350x160.
    d. These does not require output labels. These are Unsupervised learning type models.
2. Singular Value Decomposition (SVD)
    a. Singular Value Decomposition (SVD) is a fundamental mathematical technique used in linear algebra and data analysis.
    b. It factorizes a matrix into three matrices representing its singular values and left and right singular vectors.
    c. SVD has various applications, including data compression, image processing, and dimensionality reduction in methods like Principal Component Analysis (PCA).
    d. It plays a critical role in understanding and decomposing complex data structures.

Measurement Metrics for PCA Models

- **Explained Variance (Eigenvalues)**
    - This metric measures the proportion of the total variance in the data that is captured by each principal component. Higher eigenvalues indicate that the corresponding principal component explains more of the variance in the data.
- **Cumulative Explained Variance**
    - This metric provides a cumulative view of how much variance is explained as you include more principal components. It helps in selecting the number of principal components to retain while balancing dimensionality reduction and information preservation.
- **Scree Plot**
    - A graphical representation of eigenvalues (explained variance) plotted against the principal components. The "elbow" point in the scree plot is used as a heuristic to determine the optimal number of principal components to retain.

Clustering Models

3. Bayesian models
    a. Bayesian models are statistical models which checks uncertainty between different variables. Base concept used by Bayesian models is Probability.
    b. Such models can be used for Spam mail filtering.

4. Gaussian Mixture Models
    a. GMM establishes normal distributions between the data. It uses multiple normal distributions combination for it. Base statistical concept used by these models is Probability.
5. Apart from these, following are few very high performing clustering algorithms.
    a. DBSCAN (Density-Based Spatial Clustering of Applications with Noise)
    b. Hierarchical Clustering
    c. K-Means
    d. Gaussian Mixture Model (GMM)
    e. Agglomerative Clustering
    f. OPTICS (Ordering Points To Identify the Clustering Structure)
    g. Mean-Shift
    h. Spectral Clustering
    i. Affinity Propagation
    j. BIRCH (Balanced Iterative Reducing and Clustering using Hierarchies)
    k. Self-Organizing Maps (SOM)
    l. Fuzzy C-Means
    m. CLARA (Clustering Large Applications)

Measurement metrics for the Clustering models

- **Silhouette Score**
    - The silhouette score measures the quality of clustering by calculating the average distance between data points in the same cluster and the average distance between data points in different clusters. Higher silhouette scores indicate better-defined clusters.
- **Davies-Bouldin Index**
    - The Davies-Bouldin index quantifies the average similarity between each cluster and its most similar cluster. Lower values indicate better clustering.
- **Calinski-Harabasz Index (Variance Ratio Criterion)**
    - This index measures the ratio of between-cluster variance to within-cluster variance. Higher values suggest better separation of clusters.
- **Inertia (Within-Cluster Sum of Squares)**
    - Inertia measures the sum of squared distances between data points and their cluster center. Lower inertia indicates more compact clusters.

Detailed explanation for Measurement Metrics can be checked at Microsoft Documentation.

Azure Cognitive Search Services based Models

Azure Cognitive Search Services based models train themselves on a custom database. These models are connected to one or more data sources. Different types of data from connected data sources is indexed by defining a list of columns, structure the data may contain and all the data is integrated in the system.

A Search Explorer can be used to query on the underlying data, and it checks entity keywords of the questions and find the best possible matches of top N documents in from the data. A relevance score is used to decide the perfect match of a document for the given question. Azure has now integrated Semantic Search with the system, such that one can query on the search explorer using Natural Language and get the relevant information.

These cognitive search components can be integrated with GPT or other Generative AI models to curate the summarized answers from the retrieved documents. This integrated system altogether can be published as Chat based webapp with proper restriction setups.

Model Characteristics

1. These model uses pre-deployed code of Microsoft. Hence, classified under No – code or Low – Code state. Few lines of code can be there to curate the data in the required format.
2. Data Source Input format can be JSON line, JSON array, JSON format files or a SQL database table, a SharePoint location, PDFs, Images etc.
3. Data sources can be of following services.
    a. Azure Blob Storage
    b. Azure Cosmos DB
    c. Azure Data Lake Storage Gen2
    d. Azure SQL Database
    e. Azure Table Storage
    f. Azure SQL Managed Instance
    g. SQL Server on Azure Virtual Machines
    h. Azure Files (in preview)
    i. Azure MySQL (in preview)
    j. SharePoint in Microsoft 365 (in preview)
    k. Azure Cosmos DB for MongoDB (in preview)
    l. Azure Cosmos DB for Apache Gremlin (in preview)
4. Web app output is summarized answer from the retrieved document.

Deep Learning Based Models

Deep learning-based models are categorized into highly complex types of models. It tries to mimic the human brain, to learn from the unknown patterns from the input data. These models require very high processing power for computation and training. The model training can take from a few hours to multiple days based on the training data and the customization made. GPU or TPU based computers are used to train such models in lower amount of time.

Algorithms/Models in Deep Learning

LSTM (Long Short-Term Memory) model, CNN (Convolutional Neural Network), GAN (Generative Adversarial Network) are some of the examples for Deep Learning based models.

LSTM (Long Short Term Memory) Models (LSTM)

1. These models are majorly trained for sequence data.
2. Natural Language Processing task like Text sequence, timeseries sequence.
3. These models have ability to consider previous and future contexts, so, to predict next word in the line, line completion and other text processing , these models can be used.

Convolutional Neural Network (CNN)

1. CNN models are state-of-the-art models of extracting features from the data.
2. Useful to extract features from multi dimensional data or matrix.
3. Image classification, object detection, and all major deep learning models has CNN In their base.
4. These models also provides dimensionality reduction as final extracted feature map is very less in dimensions.

Generative Adversarial Network (GAN)

1. GAN models are generative types of models. Unlike CNN and other models, which classify things, GAN models generate the similar patterns based on the input data patterns.
2. Generative faces of non-existing people, generating a person's face for different age etc. are use cases for GAN.
3. GAN in the base, are modeled using two CNN models known as Generator and Discriminator.

YOLO (You Only Look Once)

1. YOLO is a popular deep learning architecture designed for real-time object detection. It stands out for its speed and efficiency in localizing and classifying objects in images or video frames, making it suitable for applications like video surveillance and autonomous vehicles.
2. OLO is known for its single forward pass approach, which means it processes an entire image through the neural network in one go, as opposed to other object detection methods

that rely on multiple passes for region proposals and classification. This design results in faster inference times.

3. YOLO has seen several iterations and variants, with YOLOv4 and YOLOv5 being notable examples. These versions have improved accuracy, speed, and additional features for specific use cases. YOLO remains a popular choice for object detection due to its real-time performance and adaptability to different applications.

Apart from these, following are very high performing deep learning models.

1. GPT-3 (Generative Pre-trained Transformer 3)
2. BERT (Bidirectional Encoder Representations from Transformers)
3. ResNet (Residual Networks)
4. Inception (GoogLeNet)
5. VGGNet
6. DenseNet
7. U-Net
8. MobileNet
9. Capsule Networks (CapsNets)

These models can train themselves on any type of complex data to find patterns from it. Following are some solutions which can be created from these models.

1. Automatic number plate recognition.
2. Face detection and recognition
3. Facial expression analysis
4. Forecasting
5. Object detection & Image classification
6. Speech to text

Model Characteristics

1. Code for the models are written in Python, C++ languages. Different environments/libraries like Tensorflow by Google, Pytorch by Facebook are used for this.
2. Input to the model can be images, video frames, numerical data, tabular data based on the requirement.
3. Output of the model can be image file with object detection, a tabular file for prediction, an audio sound etc.

Flow of Machine Learning Models

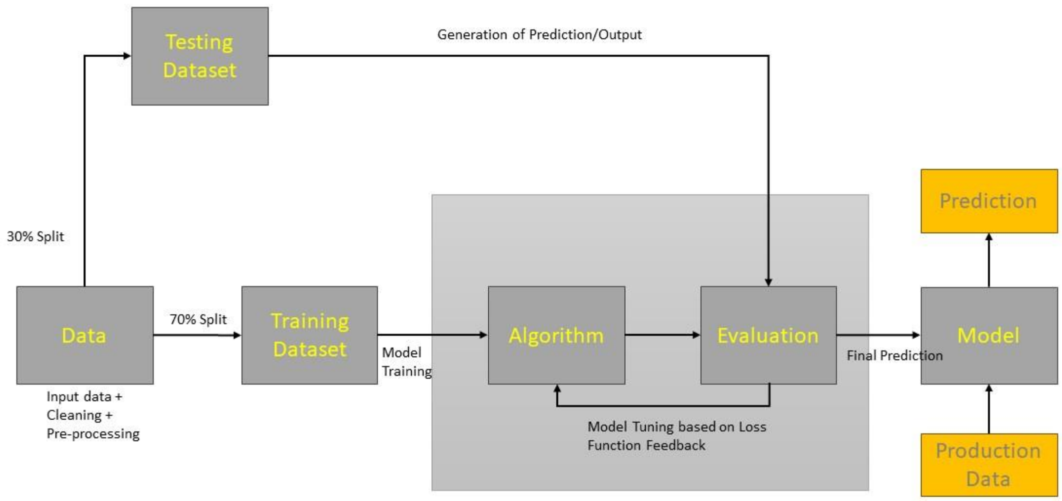


Diagram Referred from [Link]

Technologies Used to Build Statistical/ Deep Learning Models

To create effective forecasting models, you need a combination of data, tools, and expertise. Several technologies and platforms can aid in the development and deployment of these models:

1. **Azure Machine Learning (Azure ML)**

Overview: Azure ML is a cloud-based service provided by Microsoft for building, training, and deploying machine learning models, including forecasting models. It offers a range of tools and resources for data scientists and developers.

Use Cases: Azure ML can be used for building time series forecasting models, leveraging various algorithms and automated machine learning capabilities.

2. **Databricks**

Overview: Databricks is a unified analytics platform that combines data engineering, data science, and analytics. It is built on Apache Spark and is suitable for large-scale data processing and analytics.

Use Cases: Databricks is a powerful platform for processing and analyzing large datasets, making it an excellent choice for developing and deploying forecasting models that require scalable and distributed computing.

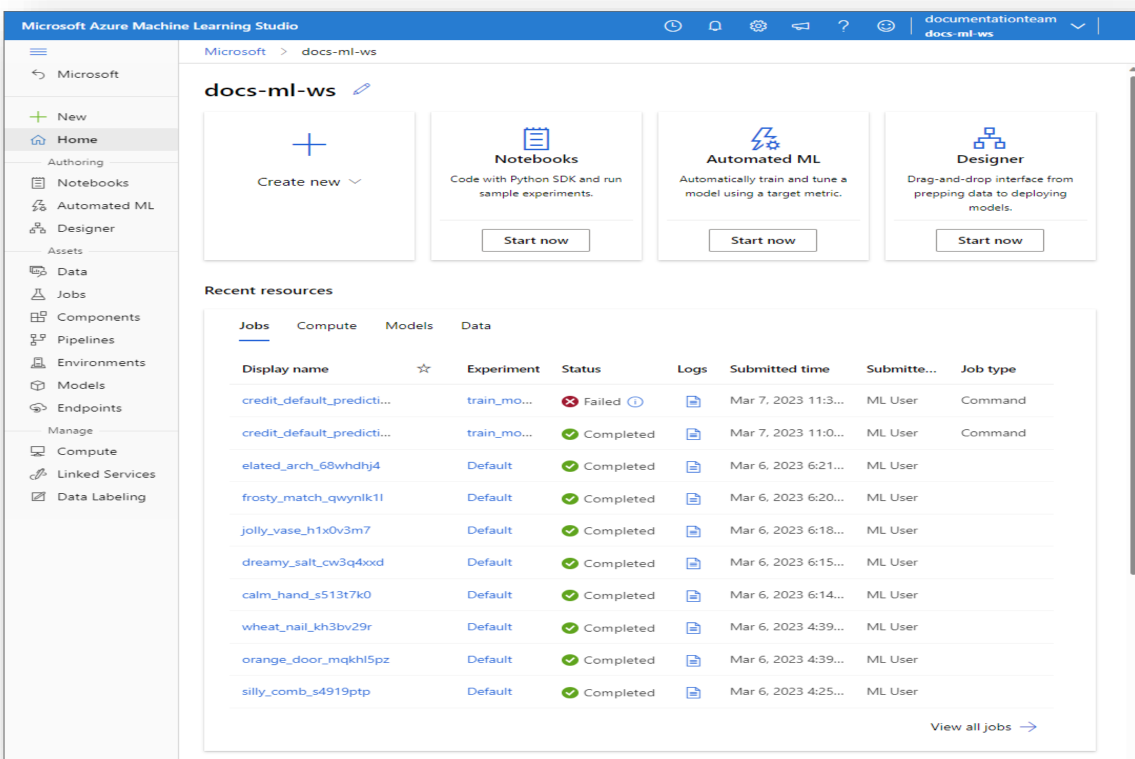
3. **Jupyter Notebooks**

Overview: Jupyter Notebooks provide an interactive environment for data exploration and analysis. They support various programming languages, including Python and R, making them a versatile choice for working on forecasting models.

Use Cases: Data scientists can use Jupyter Notebooks to develop, experiment with, and document their forecasting models.

Azure ML

Azure ML workspace or Azure ML studio provides high computational environments required to process and store different types of data, to train and deploy machine learning models from scratch. It provides support for all types complex libraries in the environment related to machine learning. Apart from that, it also provides versioning maintenance for the models



Following are the different components for Azure Machine Learning Studio.

Workspace:

1. Workspace Overview: This is your central hub for managing machine learning projects. You can access, create, and manage datasets, experiments, pipelines, and more.
2. Datastores: Datastores are connections to data storage locations, including Azure Blob Storage and Azure Data Lake Storage, allowing you to easily access and use data in your experiments.

Experiments:

1. Designer: The Designer tab allows you to create machine learning workflows visually by dragging and dropping modules to build, train, and evaluate models.
2. Notebooks: You can create and run Jupyter notebooks within Azure ML Studio, which provides a flexible environment for data exploration and model development.
3. Automated ML: Azure AutoML enables you to automatically build, train, and deploy machine learning models without extensive coding.
4. Script: This tab allows you to create and run Python or R scripts for custom model development and data processing.

Datasets:

1. Datasets: Manage and store datasets used in your machine learning experiments. You can create, import, and explore datasets here.
2. Datastores: Configure and manage datastores that connect to data sources, facilitating data ingestion and retrieval.
3. File datasets: Organize, edit, and track datasets in the workspace.

Compute:

1. Training Clusters: Create and manage compute clusters for training machine learning models at scale.
2. Inferencing Clusters: Set up and manage compute clusters for model deployment and real-time scoring.
3. Compute Instances: Provision virtual machines for interactive computing and development.

Pipelines:

1. Designer: Create, edit, and manage machine learning pipelines to automate and orchestrate end-to-end machine learning workflows.
2. Pipeline Endpoints: Deploy machine learning pipelines as REST endpoints for execution and automation.

Models:

1. Registered Models: Track and manage machine learning models you've built and registered in the workspace.

2. Designer Models: Models created within the Designer, which are used in the visual workflows.
3. AutoML Models: Models generated by Azure AutoML.
4. Models from Notebooks: Models created in Jupyter notebooks.

Endpoints:

1. Deployment Endpoints: View and manage the endpoints where your machine learning models are deployed for real-time scoring.
2. Real-time Endpoints: Manage web service endpoints for model deployment and inference.
3. Batch Endpoints: Manage batch scoring endpoints for large-scale data processing.

Conda Environments:

1. Create and manage Python environments for your machine learning projects. You can define dependencies and packages for your projects.

Notebooks:

1. Create, edit, and run Jupyter notebooks for data exploration, model development, and custom scripting.
2. Connect to your workspace and experiment resources directly from notebooks.

Automated Machine Learning:

1. Access the Automated ML interface to configure and run automated machine learning experiments.
2. View and compare automated ML experiment results.

Endpoint Experiments:

1. Monitor the performance and health of your deployed machine learning models.
2. Set up and configure data drift detection.

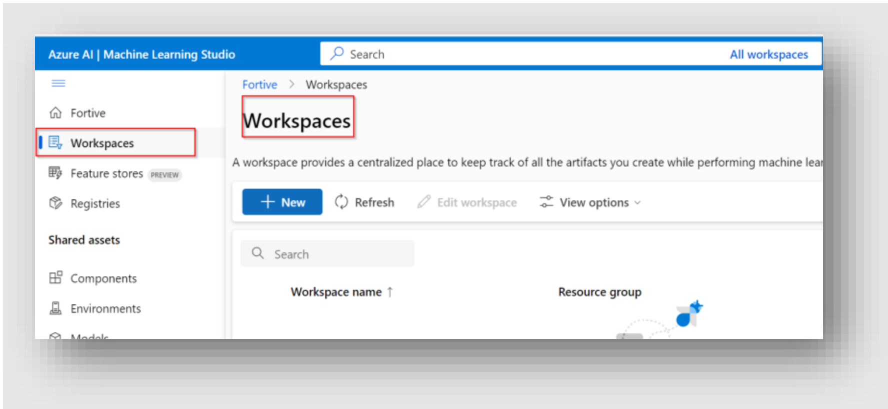
Settings:

1. Configure settings for your Azure ML Studio workspace, including workspace-wide settings and user access controls.

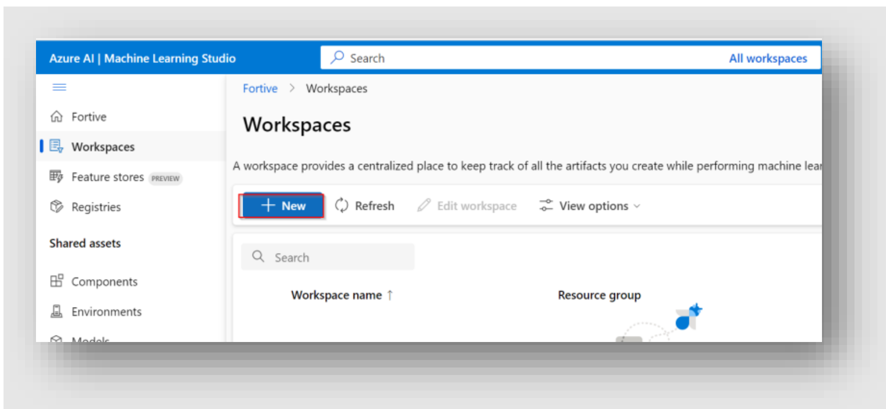
Setting up Azure ML Worksapce

Steps:

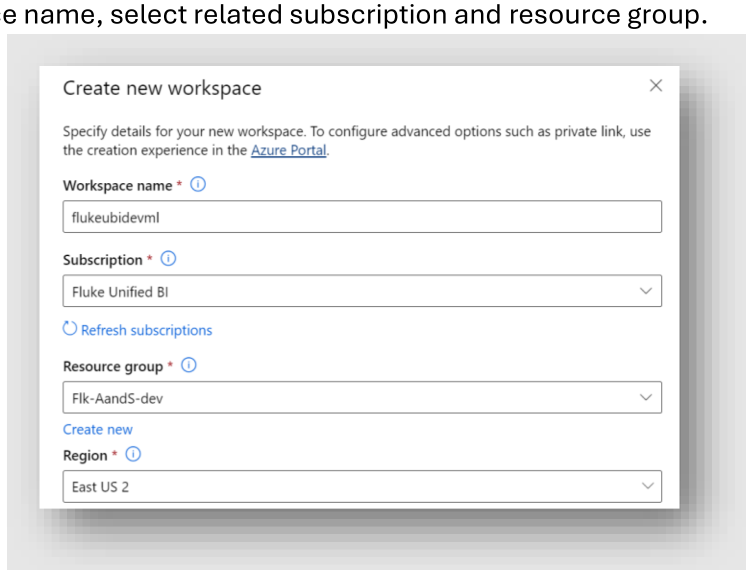
1. Go to Azure Machine Learning Studio.
2. If no workspace is created on that, it will redirect automatically a workspace creation.
3. Click on the workspace.



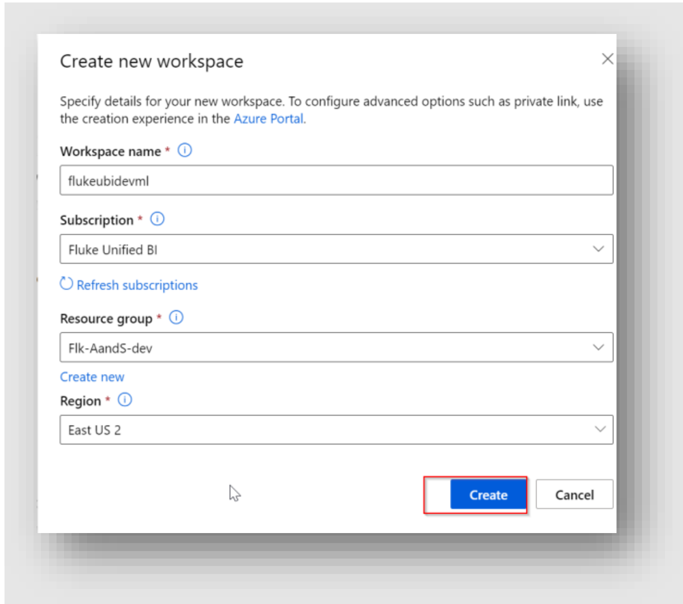
4. Select New.



5. Enter workspace name, select related subscription and resource group.



6. Select region for the resource allocation and click on Create.

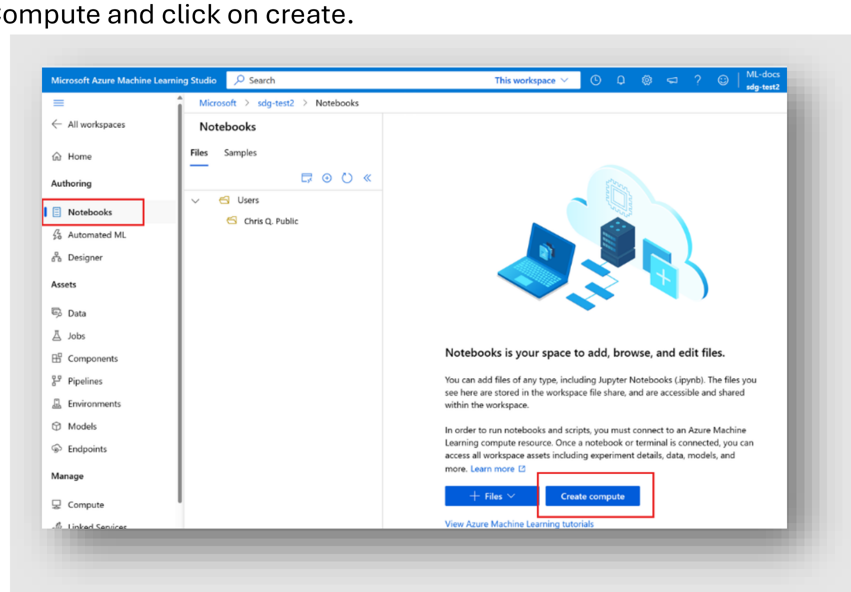


Set Up an Azure ML Compute

Compute is one more thing which is a pre-requisite to start with Azure ML.

Steps:

1. Open Azure ML Studio and Go to the created Workspace.
2. Go to the Compute and click on create.



3. Enter the details like name, subscription, and select required pricing tier/resource type.
4. Click on create.
5. Pricing of compute can be checked from the [Microsoft website.]

Pricing

1. Standard workspace creation required $9.99 /month cost, with $1 for every hour of experimentation. Ref: Link
2. Computer cost can variate based on the compute selected. The costs can be checked from the Microsoft Page
3. Resource setup: Link
4. Dev environment setup: Link

Related Blogs

1. Pricing Analysis and Optimization

Azure Data Bricks

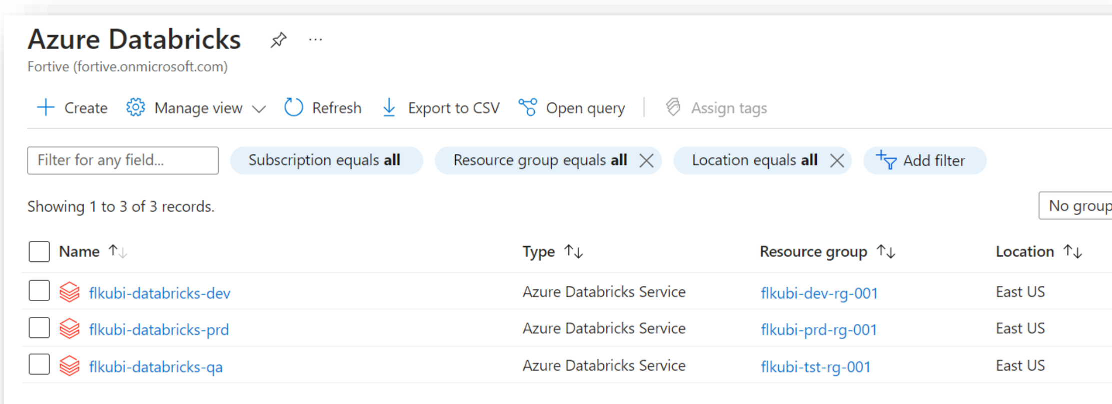
Azure Databricks is a cloud-based big data analytics and machine learning platform built on Apache Spark. It provides a collaborative environment for data scientists, data engineers, and machine learning practitioners to work on data analytics, data engineering, and machine learning tasks. Here are some key details about Azure Databricks for machine learning:

1. Unified Analytics Platform: Azure Databricks offers a unified platform that combines data engineering and data science. This means you can seamlessly transition from data preprocessing and feature engineering to model development and training.
2. Apache Spark Integration: Databricks is built on Apache Spark, which is a powerful open-source framework for big data processing and machine learning. This integration allows you to leverage the distributed computing capabilities of Spark.
3. Collaborative Workspace: Databricks provides a collaborative workspace where data scientists and engineers can work together on notebooks, experiments, and dashboards. It supports multiple programming languages, including Python, R, and Scala.
4. Machine Learning Libraries: Databricks offers pre-installed machine learning libraries like Scikit-Learn, TensorFlow, and XGBoost, making it easy to develop and train machine learning models.
5. Deep Learning: You can build and train deep learning models using popular deep learning frameworks like TensorFlow, Keras, and PyTorch.
6. AutoML: Azure Databricks supports automated machine learning (AutoML) through the use of libraries like AutoML and Hyperopt, allowing you to automate model selection and hyperparameter tuning.

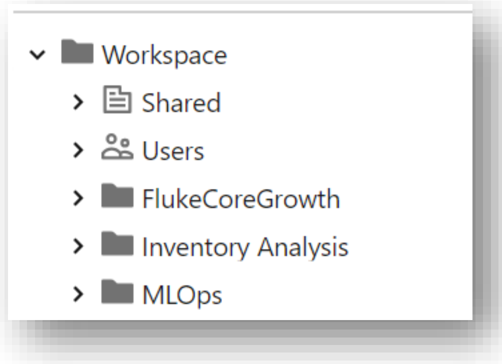
Recommendation for Setting Up Environment

Prefer Environment based resource creation.

1. For Dev, create one Databricks resource. Similarly for QA as well as for Prod create single Databricks resource for each.



2. Create workspace inside a resource on Project basis. i.e., if there are three projects, three workspace can be created.

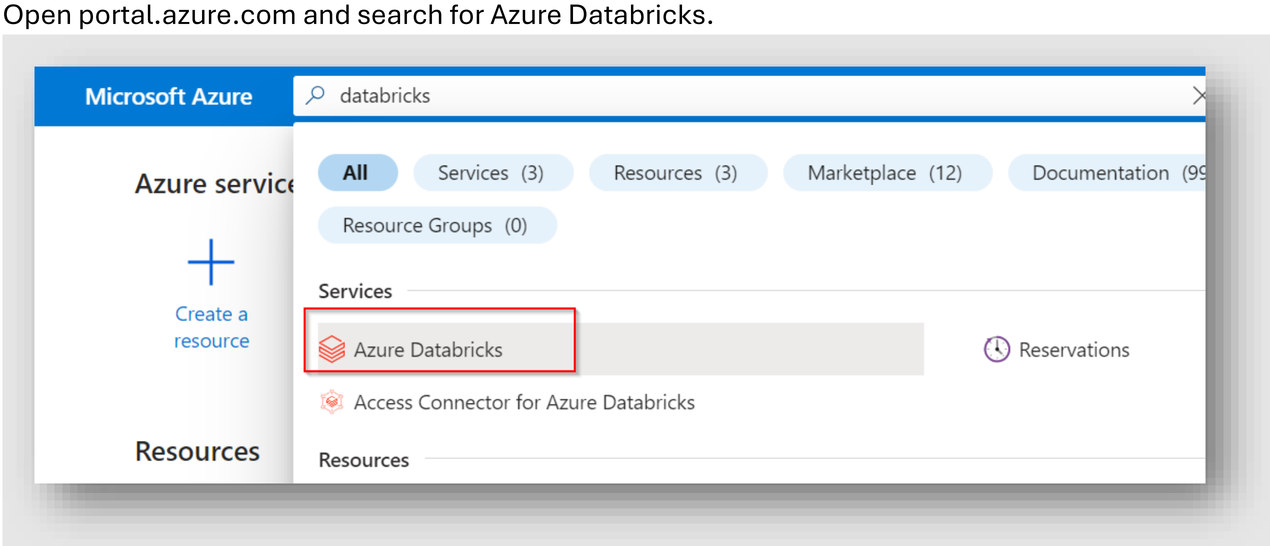


3. Maintain a proper folder structure based on the requirement inside the workspace. Separate out code for processing, code for abstract functions, code for other utilities into different folder. A hierarchy can also be created and maintained.
4. This will provide benefit for cluster sharing across workspaces. Re-use of utilities across workspace in all the projects.

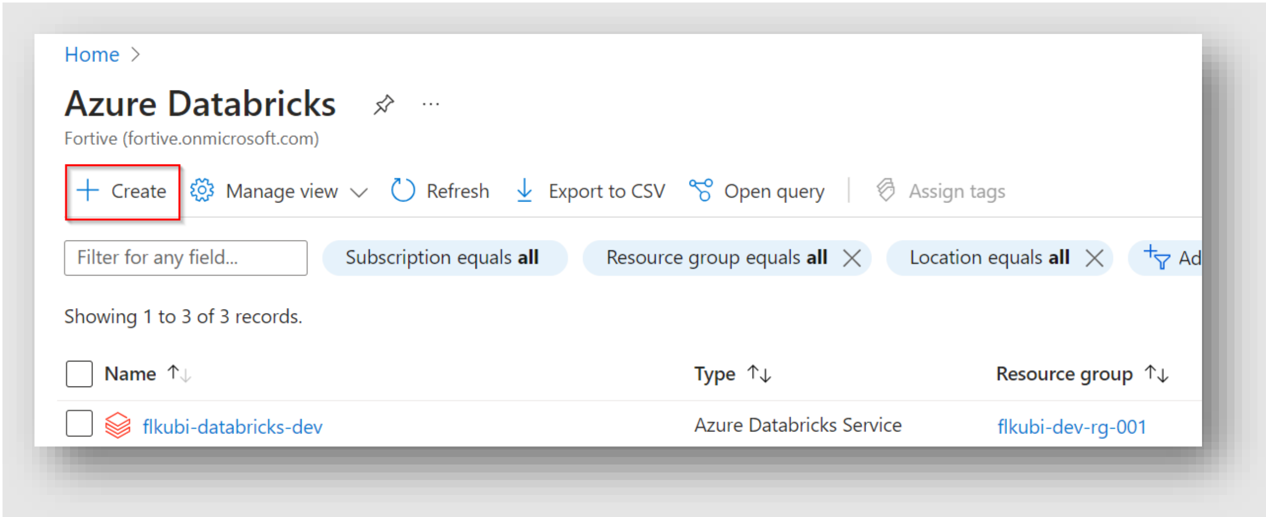
Set up Azure Databricks Workspace

Steps

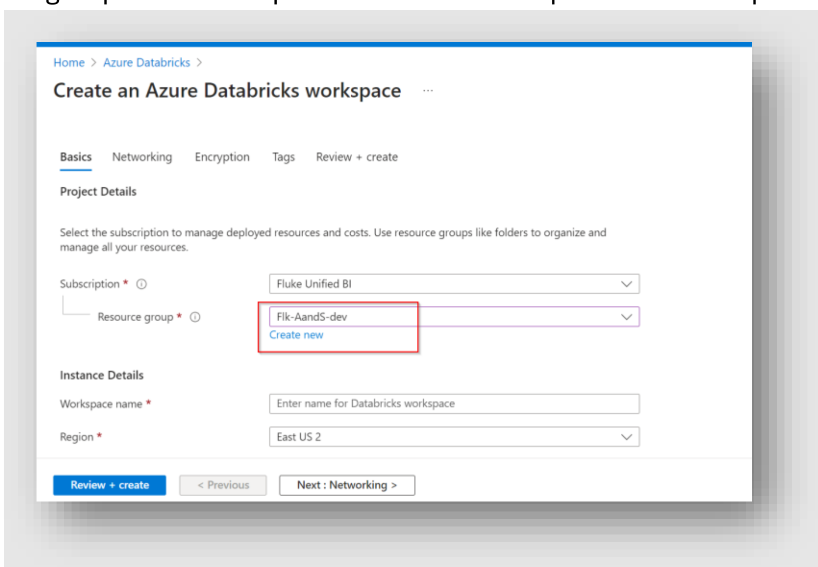
1. Open portal.azure.com and search for Azure Databricks.

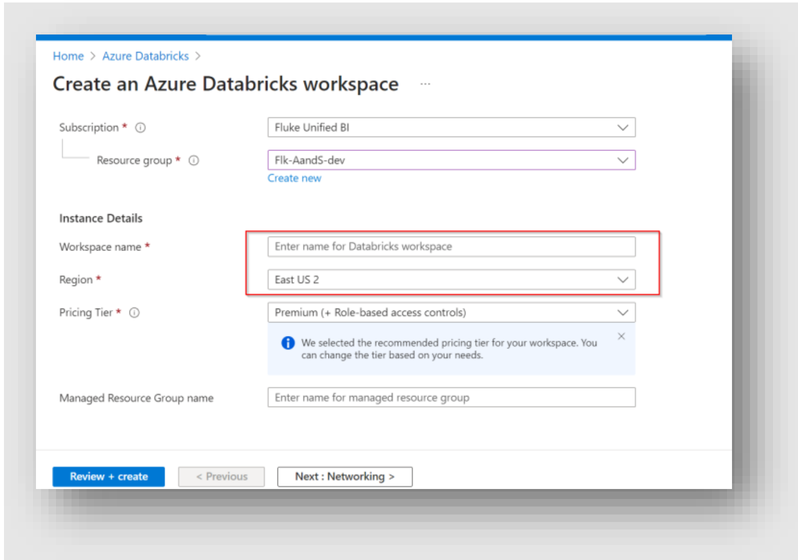


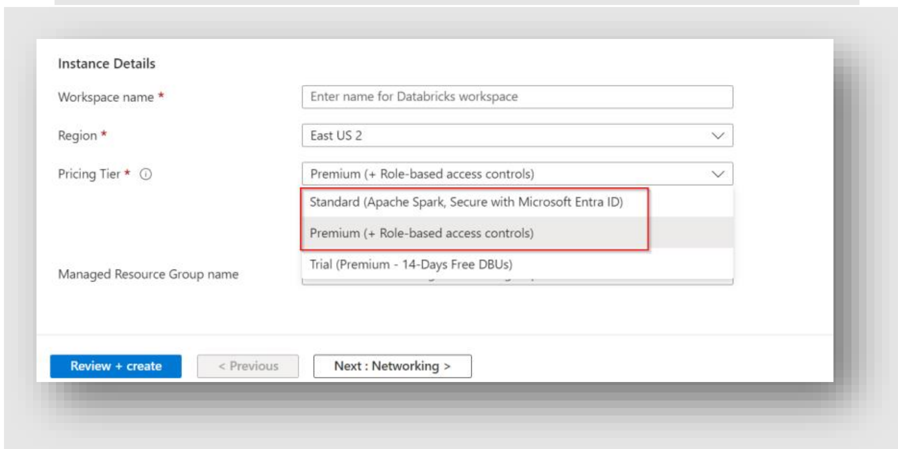
2. Click on Create to create an environment.



3. Select resource group and subscription. Provide a workspace name and pricing tier.







Pricing

1. Compute pricing tiers can be check from the Microsoft documentation for Databricks.

Azure Databricks Vs. Azure ML

Data Distribution

1. Azure ML provides environment and compute to train, deploy, maintain versioning for ML models. Azure ML Notebooks are good when you are training with a limited data on single machine. Though Azure ML provides training clusters, the data distribution among the nodes is to be handled in the code.
2. Azure Databricks with its RDDs are designed to handle data distributed on multiple nodes. This is advantageous when your data size is huge. When your data size is small and can fit in a scaled up single machine/ you are using a pandas DataFrame, then use of Azure DataBricks is a overkill

Data Cleaning

1. Azure Databricks, with its native support for various file formats and integration with Apache Spark, makes data cleaning and manipulation more straightforward and efficient for handling large datasets. In contrast, Azure ML notebooks may require more custom coding and data handling for similar data cleaning tasks, which can be less convenient, especially when dealing with extensive and diverse datasets.

Training Both has the capabilities if distributing the training, Databricks provides inbuilt ML algorithms that can act on chunk of data on that node and coordinate with other nodes. Though this can be done on both AzureMachineLearning and Databricks with tf,horovod etc.

If data is large then Databricks is good option, otherwise choose Azure ML.

Although, **best way** to utilize these is to have data processing done in Azure Databricks, handle model versioning and deployment with other automation in the Azure ML workspace.

Reference: Link, Link

Azure AI Cognitive Search Services Based Models

Cognitive search-based models represent a powerful paradigm shift in the way we interact with and extract value from vast amounts of unstructured data. These models integrate the capabilities of traditional search engines with artificial intelligence (AI) and natural language processing (NLP), resulting in intelligent, context-aware, and highly efficient search experiences.

Cognitive search-based models address these limitations by going beyond mere keyword matching. They enhance search capabilities in the following ways:

1. Semantic Understanding: Cognitive search models employ NLP techniques to understand the meaning and context of words, phrases, and documents. This enables users to find information based on intent rather than just specific keywords.
2. Entity Recognition: They identify and extract entities like names, dates, and locations from documents, allowing for more precise and relevant search results.
3. Relevance Scoring: Cognitive search employs relevance scoring to rank search results based on their significance to the user's query, delivering more accurate results.
4. Personalization: Cognitive search considers user behavior and preferences, offering personalized search results and recommendations.
5. Faceted Search: Users can filter and refine search results using facets, making it easy to navigate through large datasets.
6. Question Answering: Some models support natural language questions, providing direct and context-aware answers from documents.
7. Visual Search: In addition to text-based search, cognitive search can extend its capabilities to images and videos, allowing users to find information within multimedia content.

List of Resources

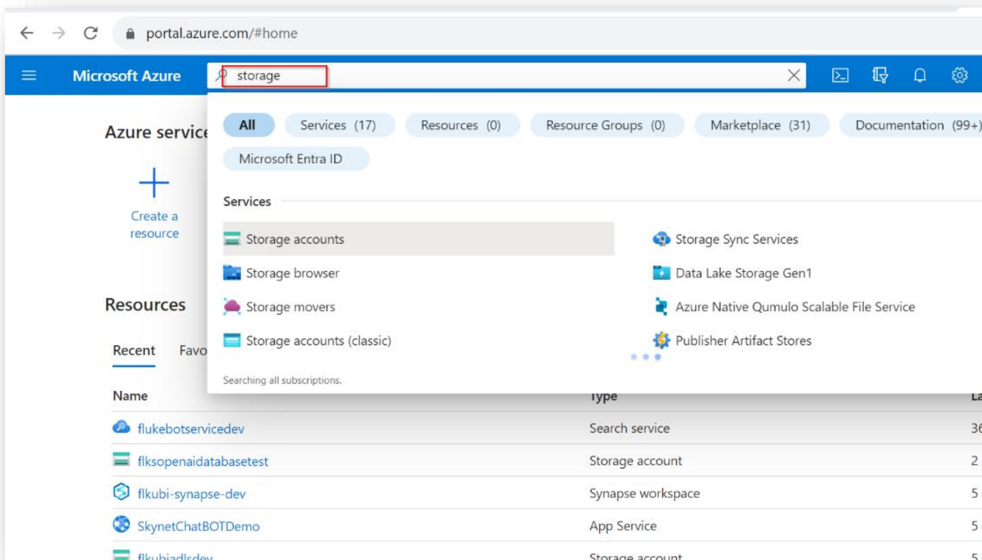
The following are the resources which need to be setup/deploy while creating the a Azure AI BOT with Cognitive search and Azure Open AI integration.

1. Data Sources (ADLS/Azure SQL DB etc.)
2. Azure Cognitive Search Resource
3. Azure Open AI Resource
4. Web App

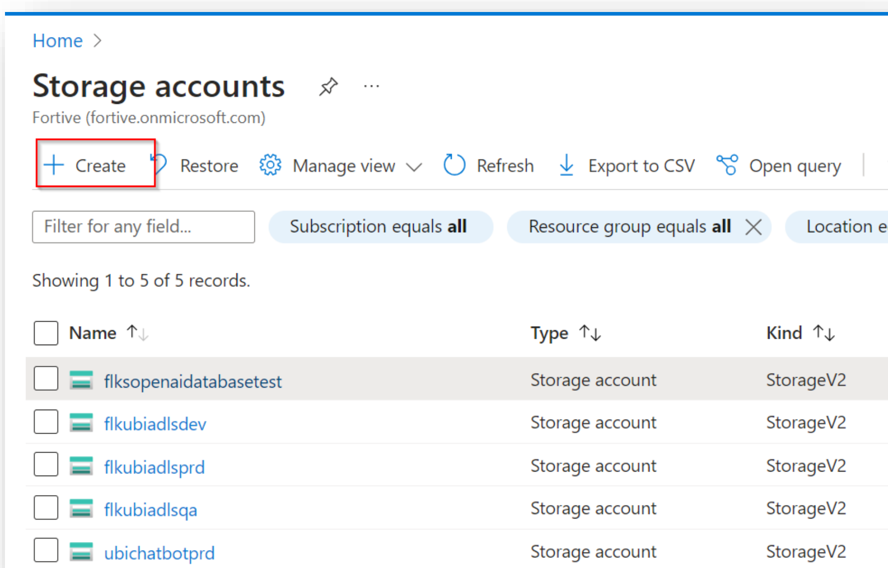
Data Storage

Set up Storage Account

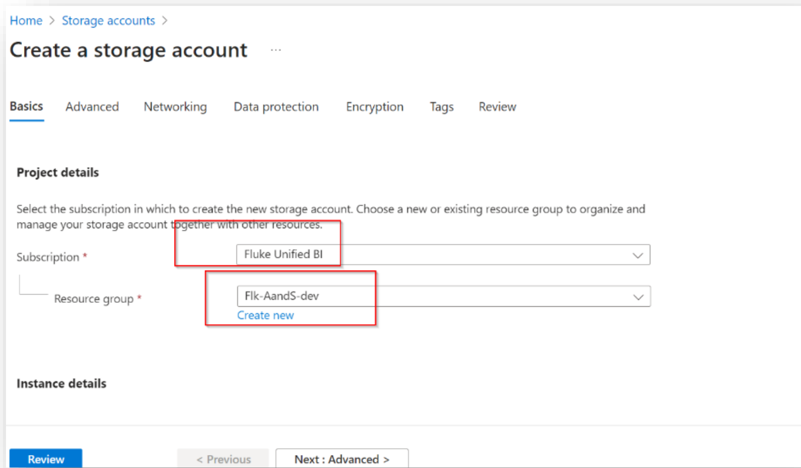
1. **Login to portal.azure.com with valid credentials and search for storage accounts.**

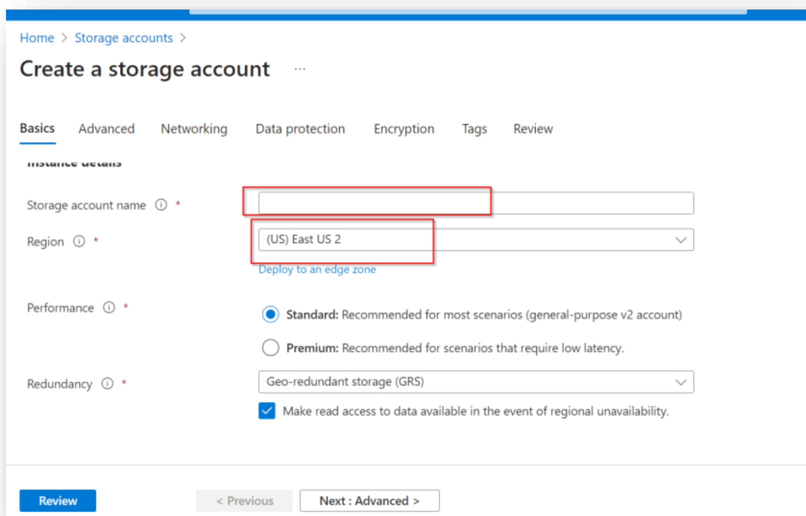


2. Click on create for creation of a storage account.

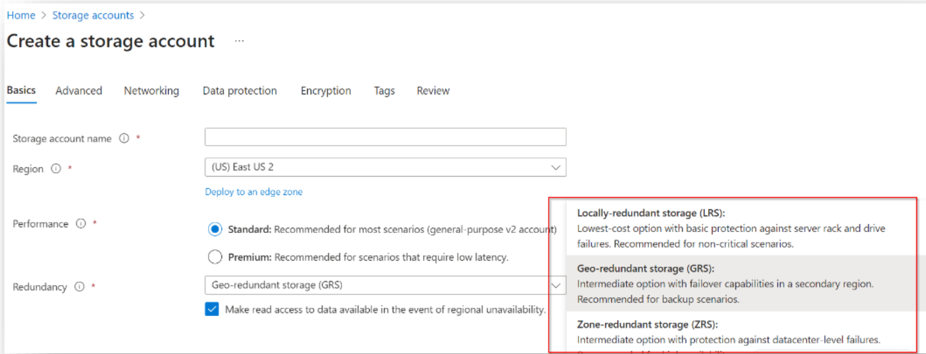


3. Select subscription, resource group, name and pricing tiers.

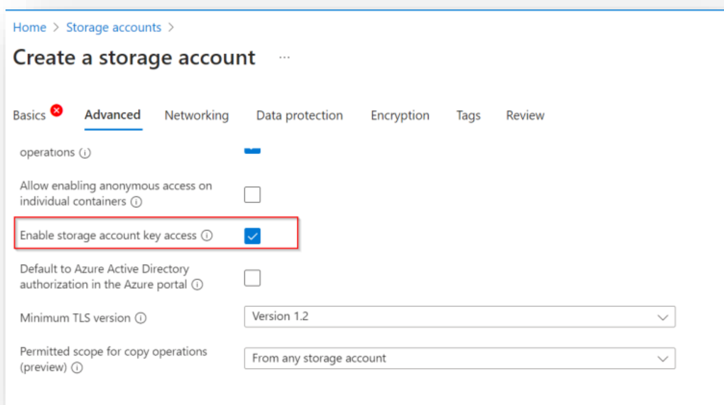


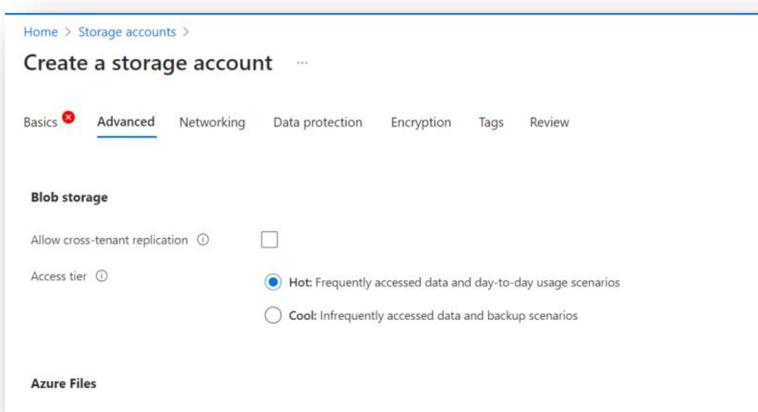


4. Select performance, region and redundancy options
    a. Standard performance, as ADLS frequent access for Open AI model is not required.
    b. Region: Select proper region considering the customer base.
    c. Redundancy: Redundancy consist of different strategies to employ in case of server's unavailability or failure. As Microsoft maintains different versions of data storage, redundancy requires selection of proper strategy to implement it.



1. Select the advanced options such as Hot/Cold storage, ACL, enabling hierarchical namespace (feature of ADLS Gen2) provided for ACL. Here, for Cognitive Search resource, indexer runs at specific time to integrate further data. So, if that is the case then Hot access tier is preferred.
    a. Meanwhile, access via ADLS key option should be enabled





Azure Cognitive Search

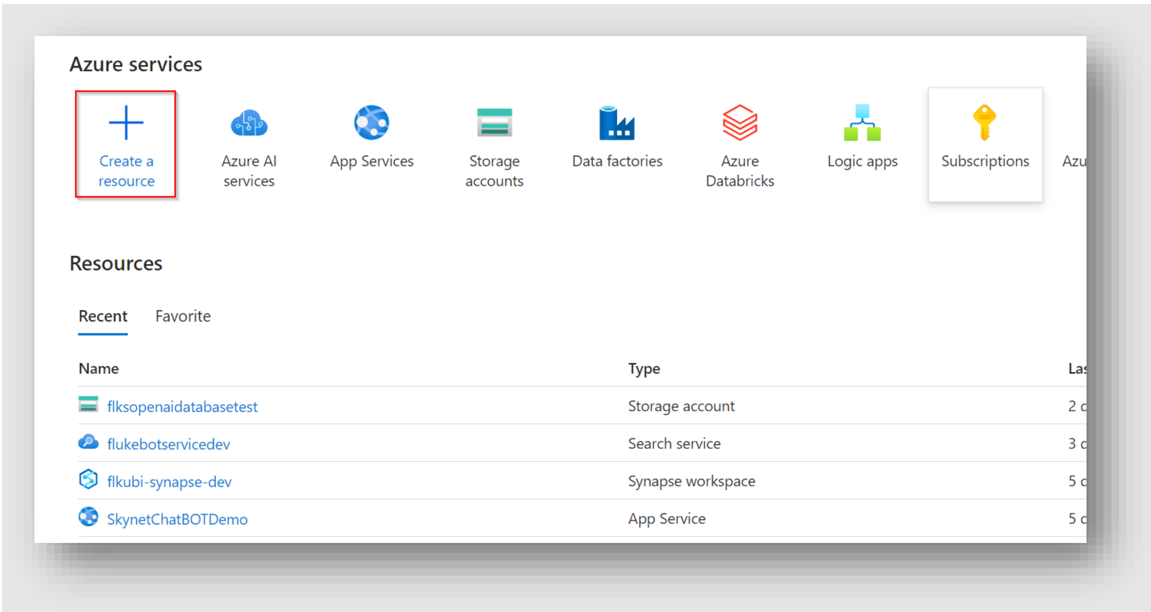
Azure cognitive search is a service provided by Microsoft which indexes a custom Data source having specific type of data. This indexing mechanism is further used to search through the documents/records. The following are the terminologies associated with Azure Cognitive Search.

1. Index: An index is a structured representation of data used for efficient searching. It defines how data is organized, which fields are searchable, and how the search results are ranked.
2. Indexer: An indexer is a component responsible for extracting data from external sources, such as databases or content repositories, and populating an Azure Cognitive Search index with that data.
3. Data Source: A data source is the origin of the data you want to search. It can be a database, Azure Blob Storage, or other external repositories where your content is stored.
4. Skillset: A skillset defines a series of AI-based operations applied to the data during indexing, enabling data enrichment, transformations, and entity recognition.
5. Skill: A skill is an individual operation within a skillset, such as text extraction, translation, or image analysis, used to process and enhance the content during indexing.

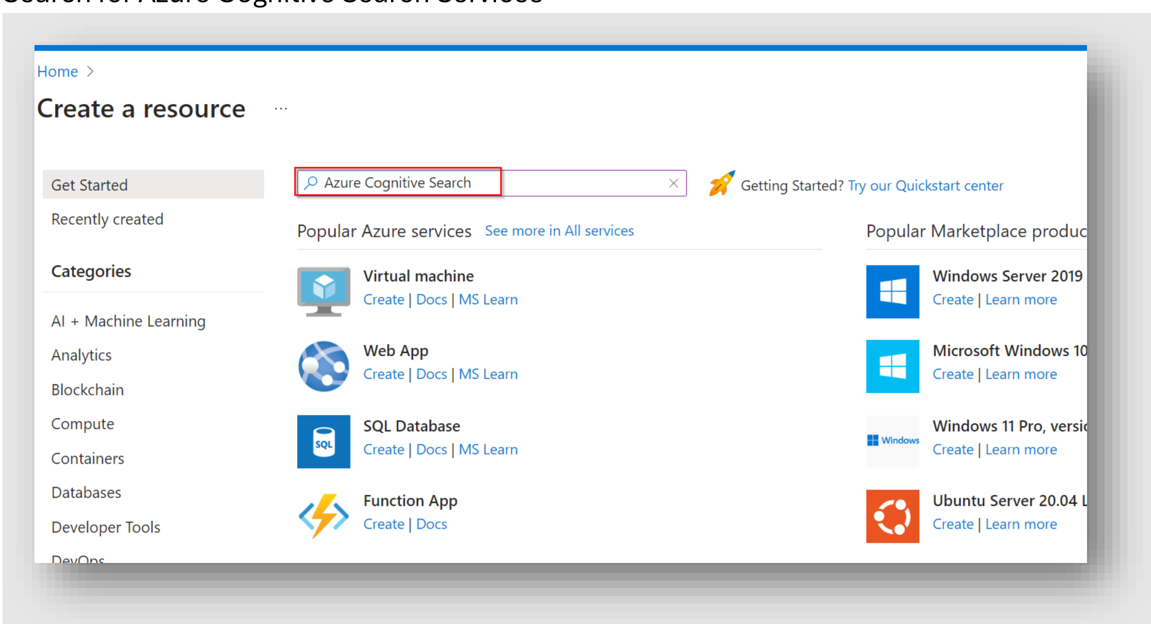
Set up Azure Cognitive Search Services

Basic Steps

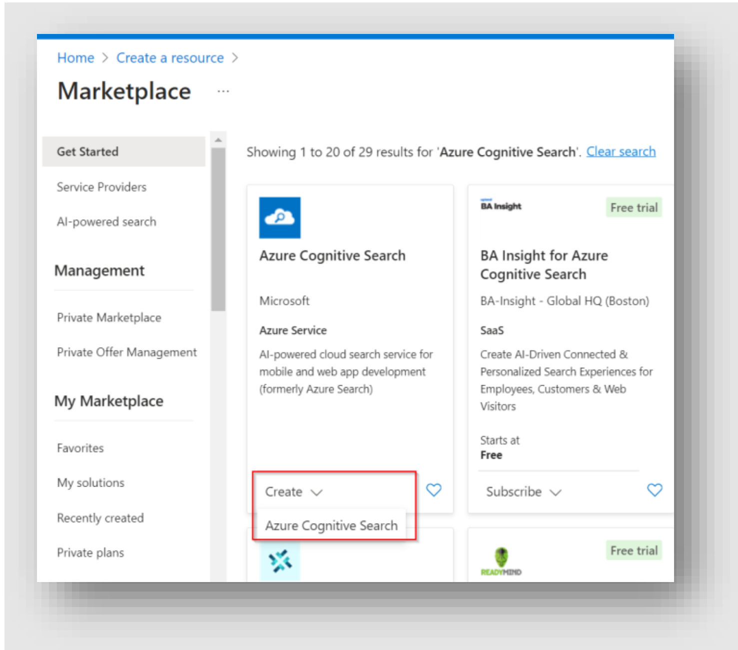
1. Log in to the Azure portal.



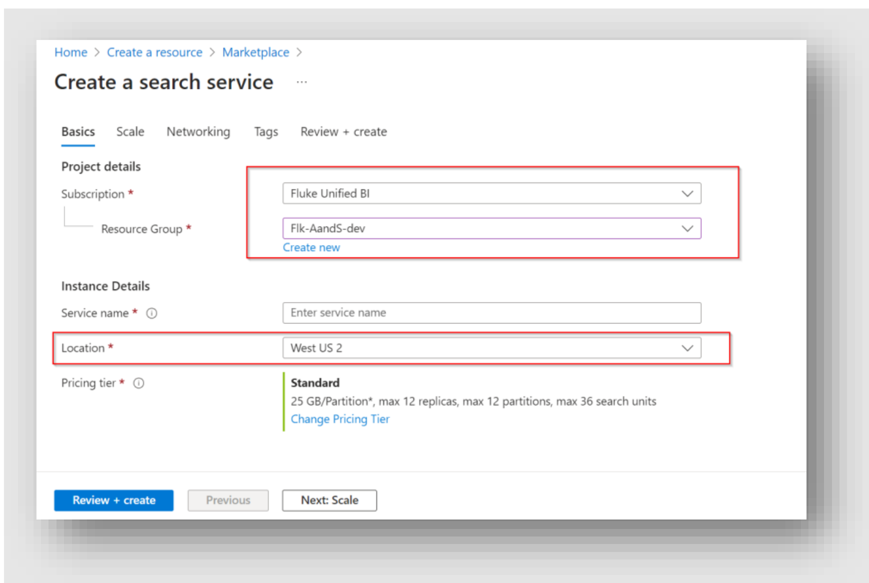
2. Search for Azure Cognitive Search Services



3. Create a Search service.

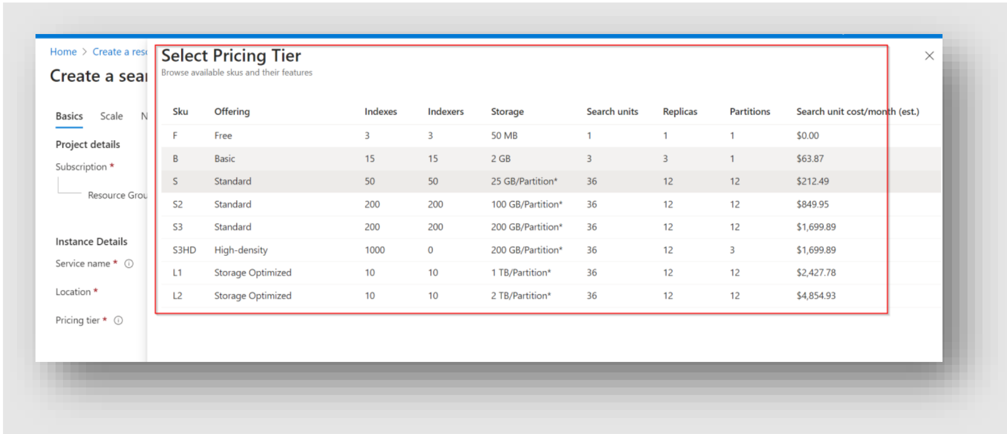


4. Select a name and location.



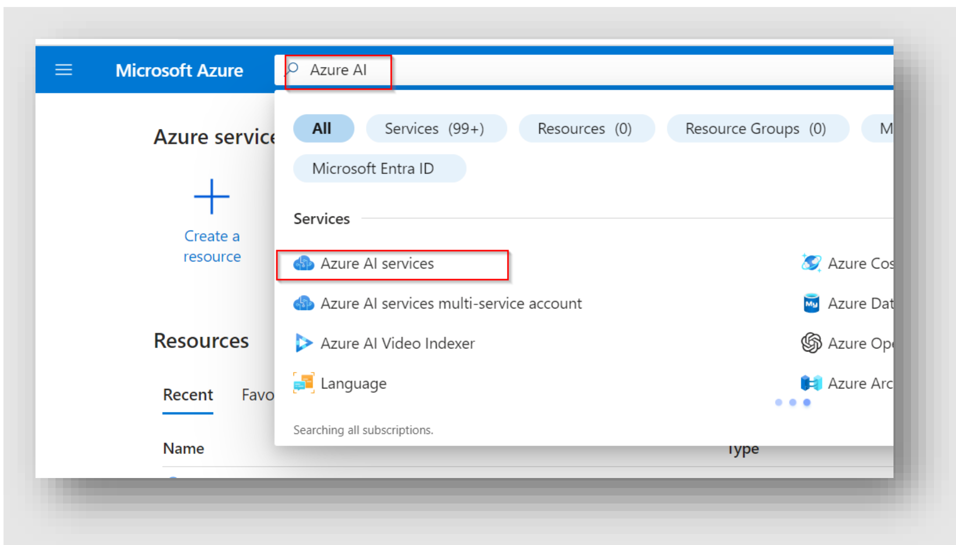
5. Change the pricing tier if you want. Most of the cases, 25 GB Per Partition * 12 = 300GB is enough for most of the data. If the data is so huge that it requires more memory than that then higher pricing tier can be chosen.

   Free pricing tier is available for trial purposes. For experimentation purposes, it is optimal to choose.

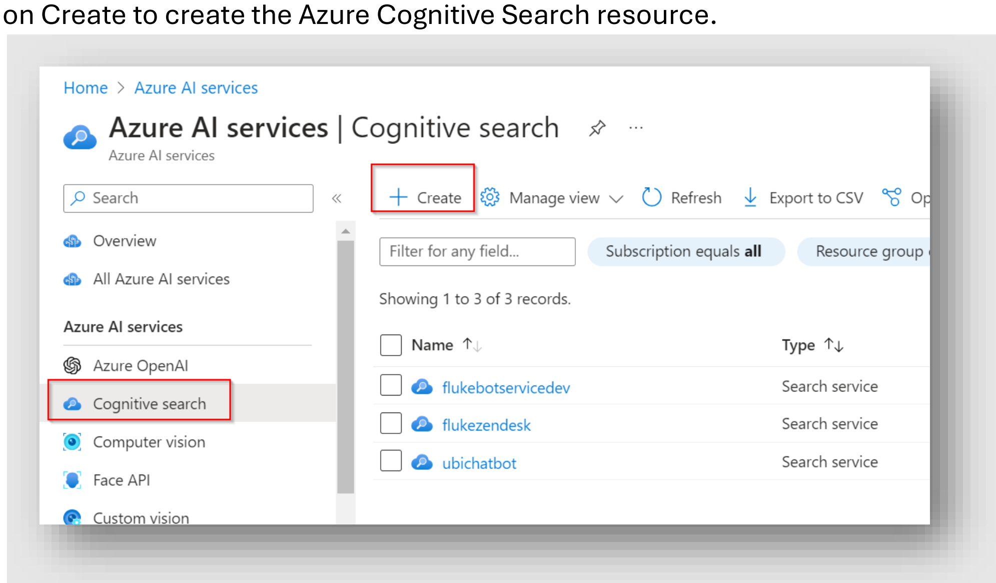


Alternatively, Azure Cognitive Search can directly be created from Azure AI services

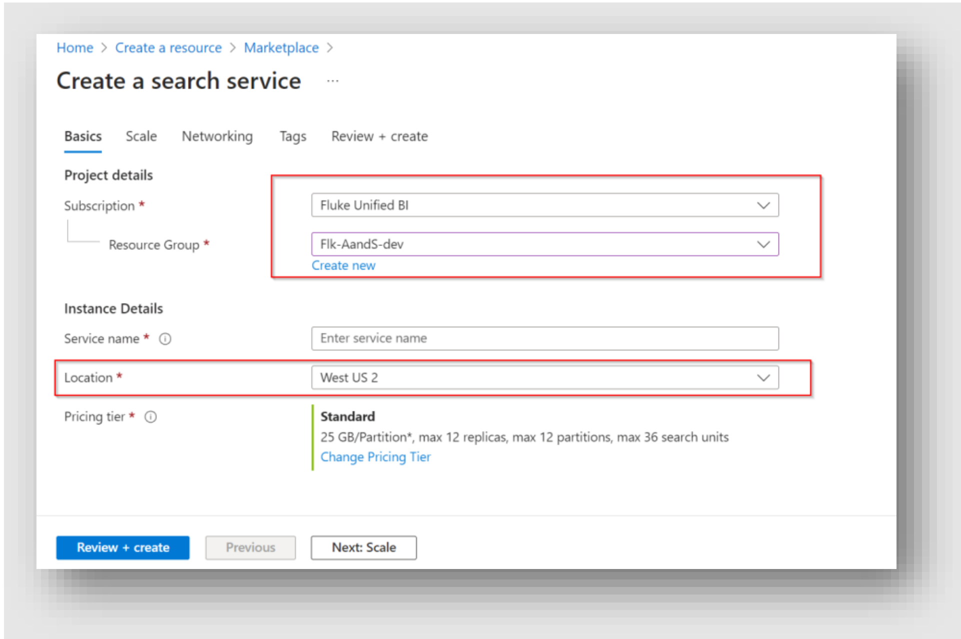
1. Select the Azure AI services resource.



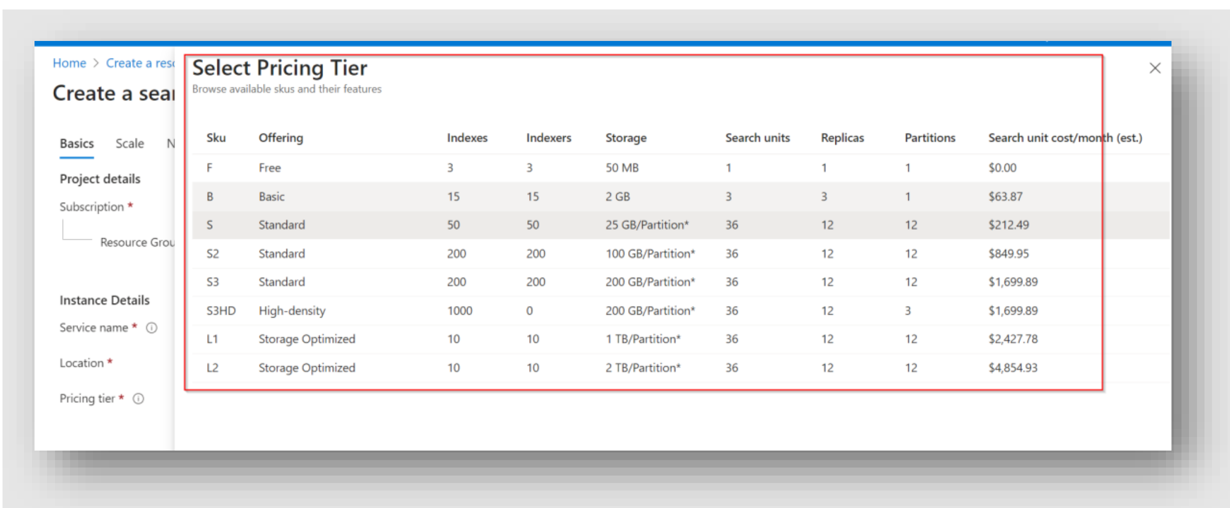
2. Click on Create to create the Azure Cognitive Search resource.



3. Select a name and location.

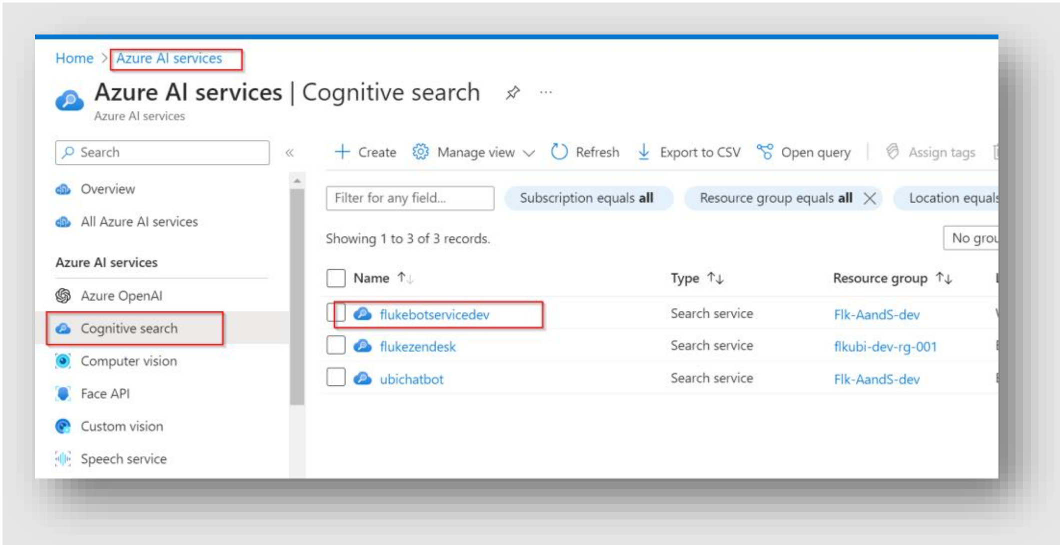


4. Change the pricing tier if you want. Most of the cases, 25 GB Per Partition * 12 = 300GB is enough for most of the data. If the data is so huge that it requires more memory than that then higher pricing tier can be chosen.
Free pricing tier is available for trial purposes. For experimentation purposes, it is optimal to choose.



Verification:

Once the resource is created, it should be shown in Azure AI Service's Cognitive Search window.



Reference Links

1. **MS Learn: Link**

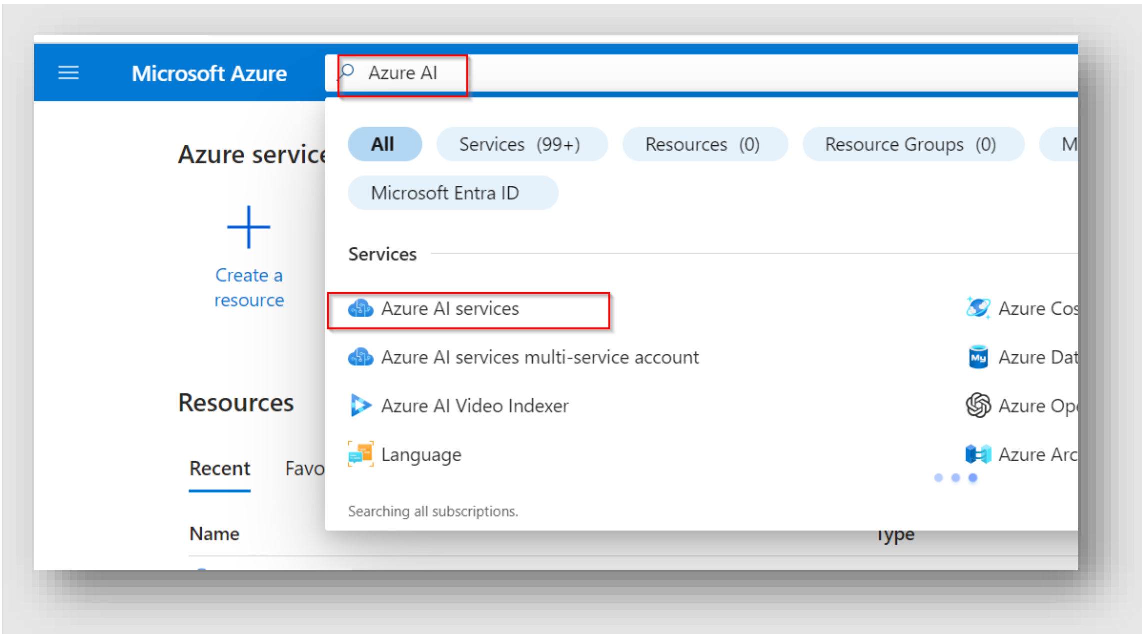
Reference Blogs:

1. **Create Azure Cognitive Search Services: Link**

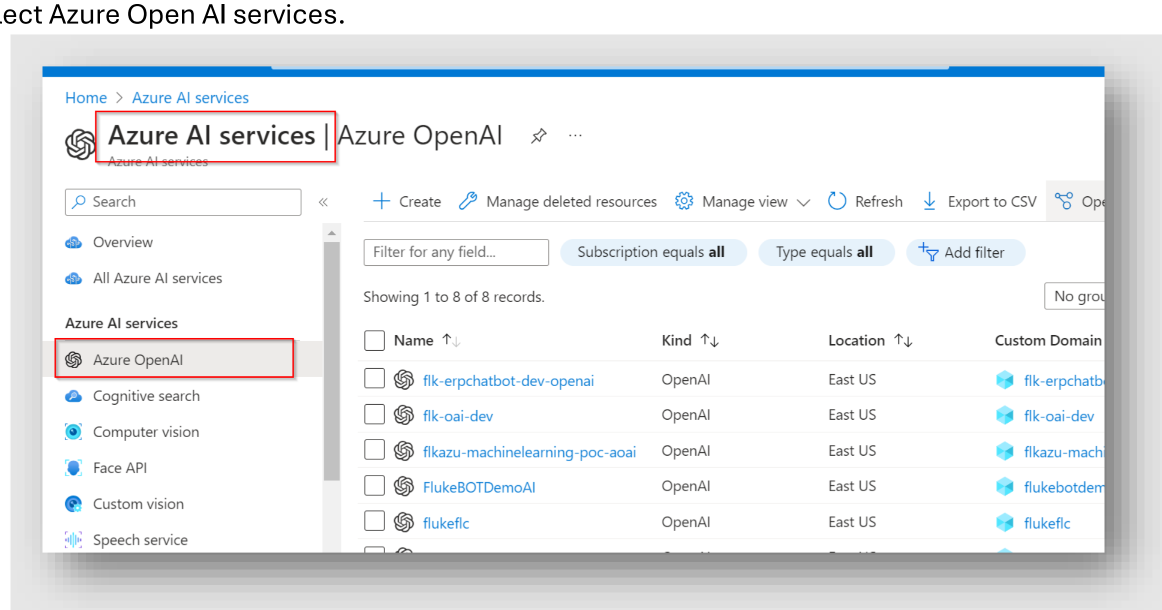
Setup/Deploy Up Azure Open AI Resource

Steps:

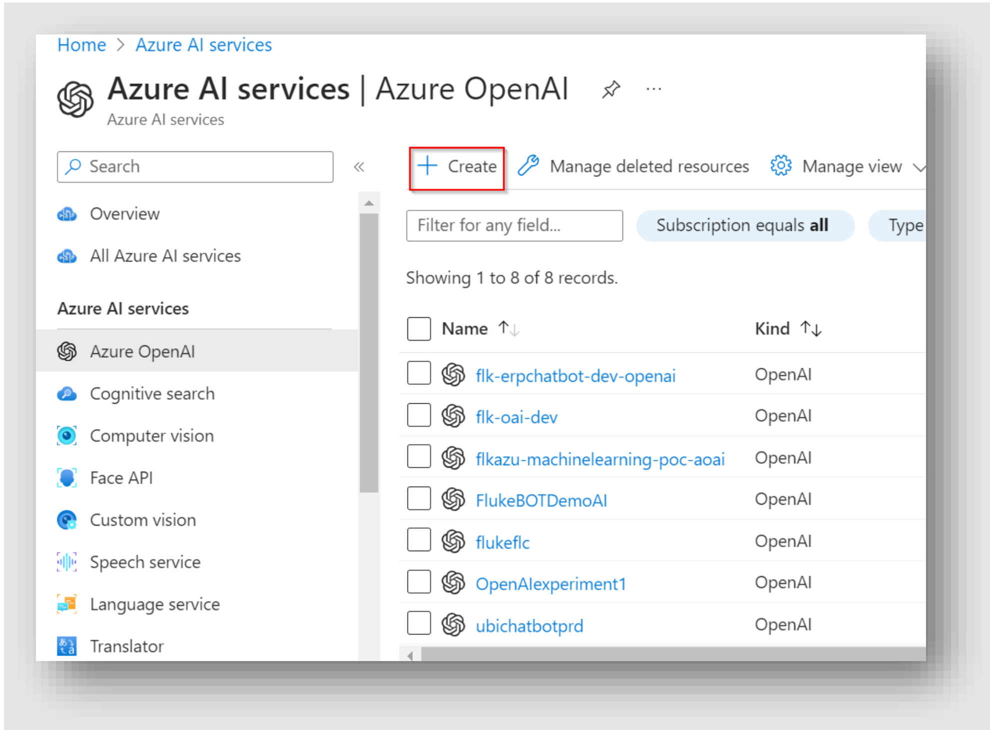
2. Select the Azure AI services resource.



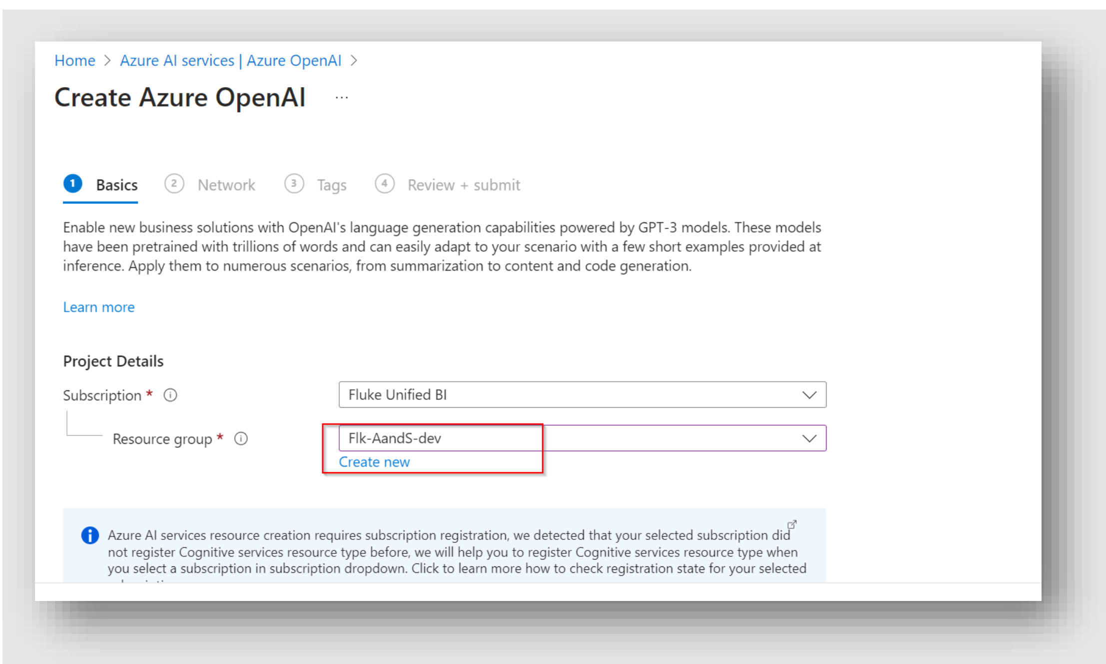
3. Select Azure Open AI services.

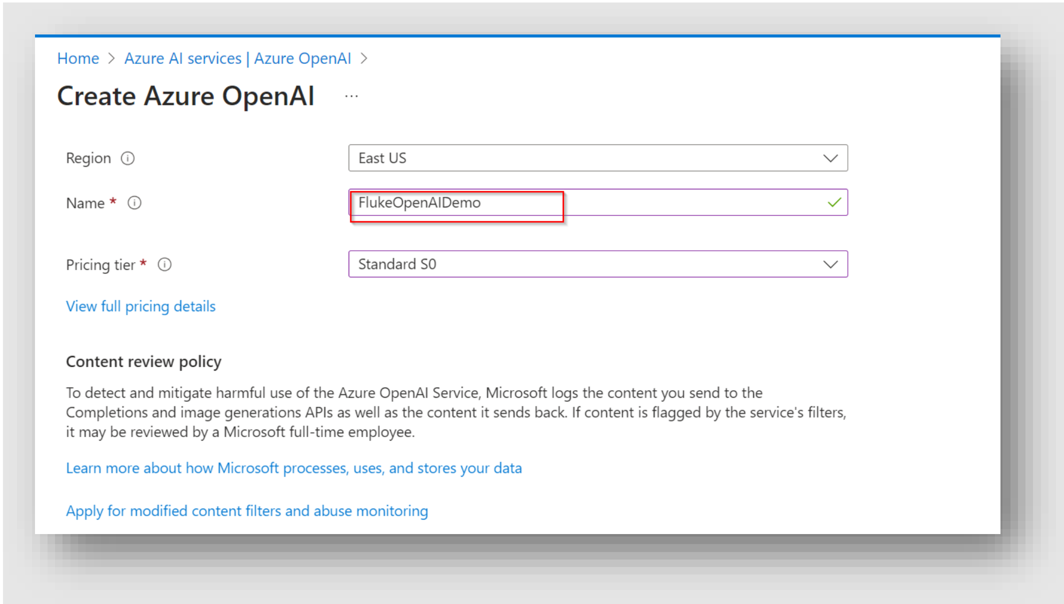


4. Click on create to for model selection and deployment.

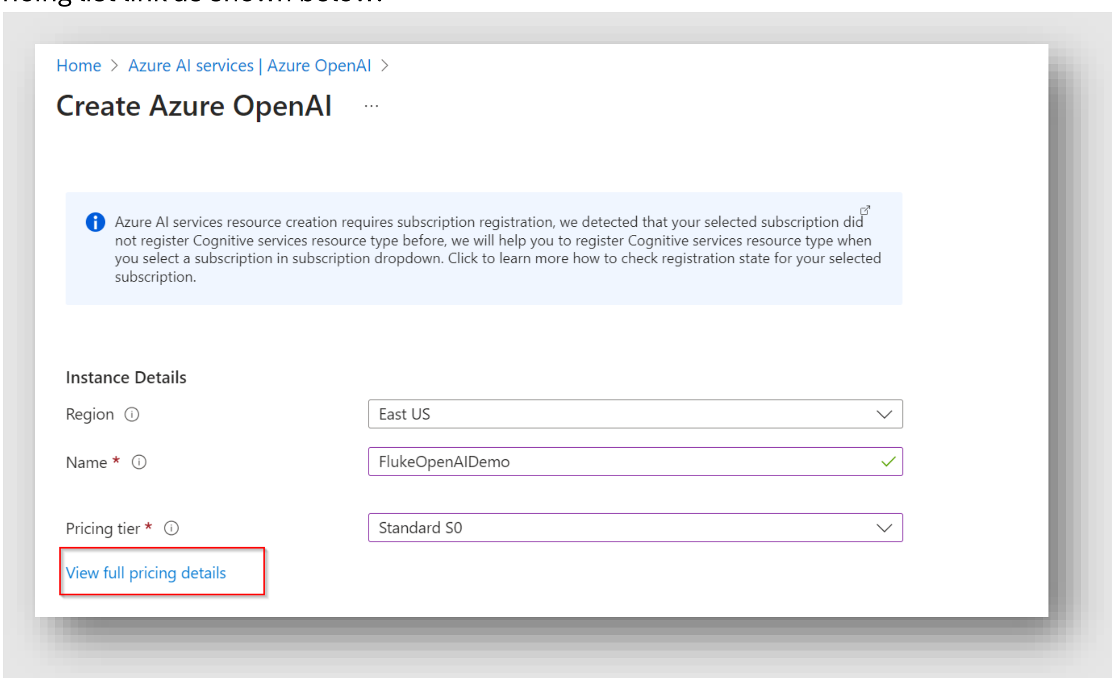


5. Select resource group, name and pricing tiers.





6. Change the pricing based on the requirements. View the detailed pricing tiers by clicking on the full pricing list link as shown below.



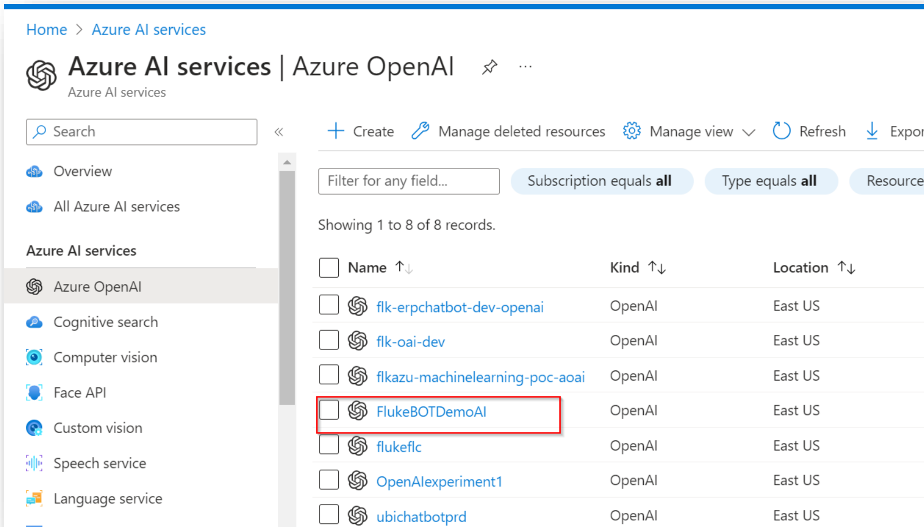
7. Alternatively, pricing tiers can be seen from the [Azure Open AI Pricing Tiers](). It supports different types of NLU LLM models like GPT 3.5, GPT 4, Babbage, DaVinci etc.

References

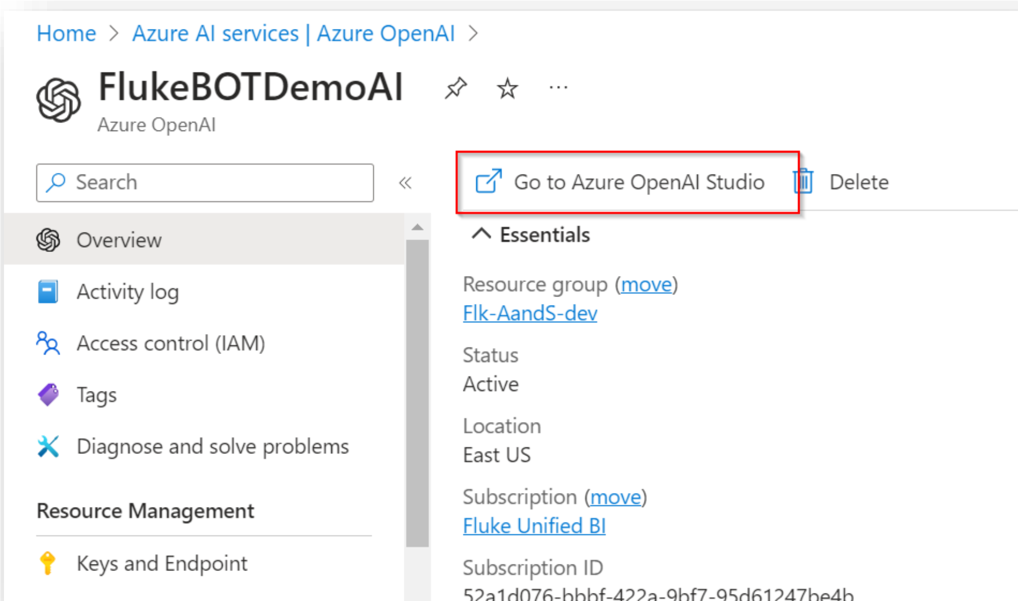
MS Learn: Link

Deploy a GPT Model in Azure Open AI Resource

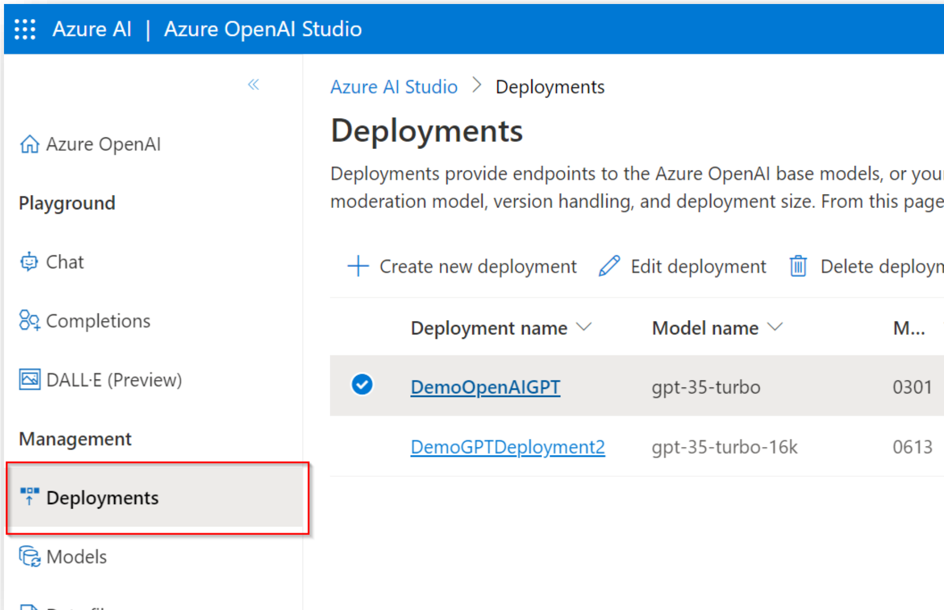
1. Go to Azure Open AI services and select the relevant Azure Open AI resource



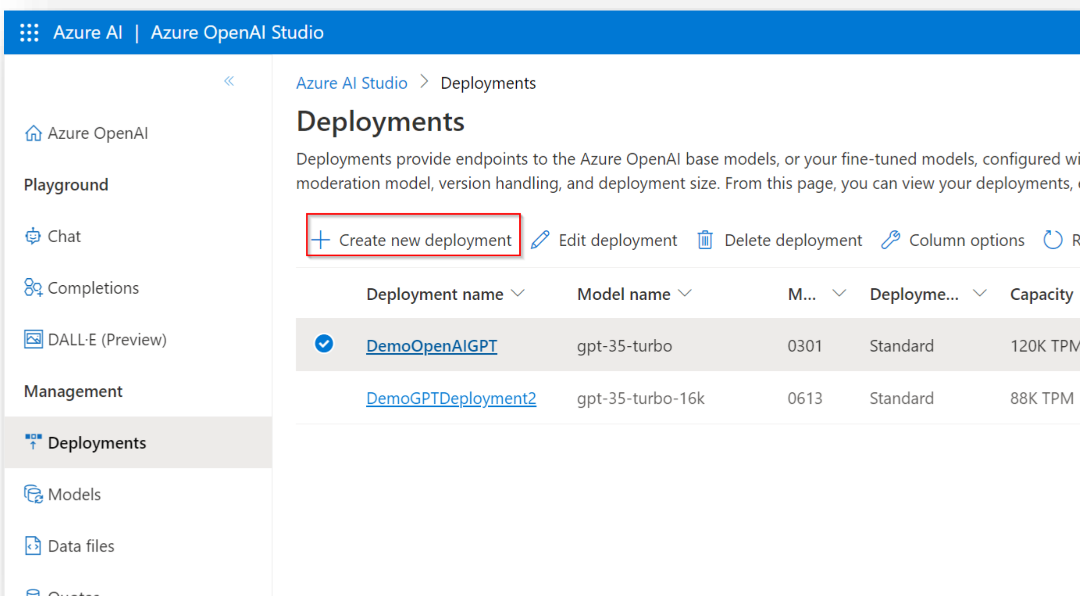
2. Now click on the button to open Azure Open AI studio.



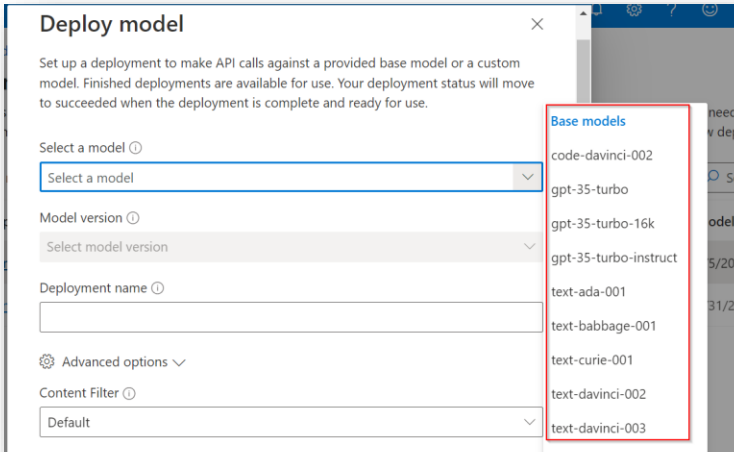
3. Now in the Open AI studio, select the deployments.



4. Create a new Deployment.



5. Select specific model version to be deployed and provide a relevant name to it.



6. Click create.

Different Automation Possibilities

1. Indexing SharePoint Files
    a. Currently SharePoint can not be used as a direct data source to Azure Cognitive Search Services. Automation setup is required to copy SharePoint files to available DataSource options.
    b. Copy Azure SharePoint to Blob/ADLS Storage
        i. Power Automate flows can be used to setup copy activity automation to copy the SharePoint files to blob/ADLS storage.
        ii. Power Automate can be used to setup the automation for copying files from Azure SharePoint to ADLS.
        iii. An create/modify file trigger can be setup on the files to copy the updated, newly added files to blob storage automatically.
    c. Copying files using Scripts.
        i. Purview data can be copied as a CSV format to ADLS/Blob storage using a python script code.
        ii. Python script can be hosted in Azure Databricks or another cloud notebook environment and can be scheduled to run on specific frequency to copy the latest catalog data to blob storage.

Reference Links

1. ARIMA Model: Link
2. SARIMA Model: Link
3. Supervised and Unsupervised Models: Link
4. Principal Component Analysis: Link
5. Support Vector Machine (SVM): Link
6. Singular Value Decomposition (SVD): Link
7. Reinforcement Learning: Link
8. Evaluate Classification Models: Link
9. Evaluate Regression Models: Link
10. Generative Adversarial Networks: Link
11. Convolutional Neural Network: Link
12. Deep Learning Models: Link
13. Azure ML Studio: Link
14. Simple Regression: Link
15. Logistic Regression: Link
16. Clustering Algorithms: Link
17. DB Scan Algorithm: Link