

# **One Month Project On**

# **MLOPS Using Azure Cloud**

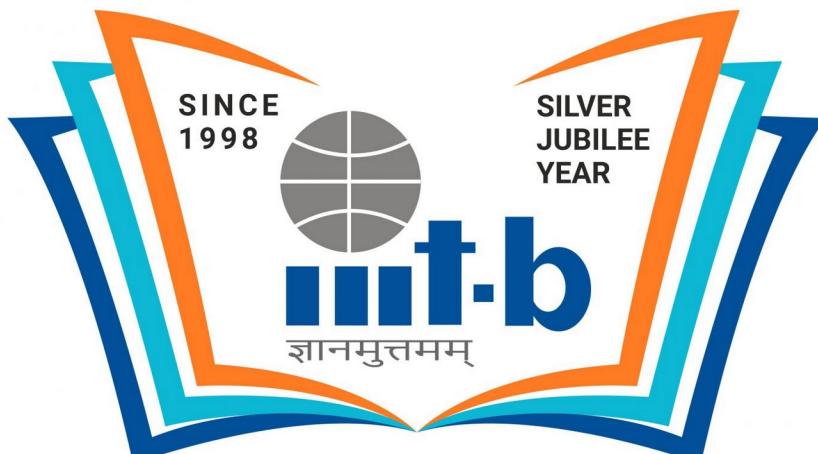
**Submitted by :-**

**Harsh Tripathi (MT2022048)**

**Vipul Manohar Ahuja (MT2022132)**

**Under Guidance of:-**

**Prof. B. Thangaraju**



**Pioneering Excellence in Education,  
Research & Innovation**

## **Problem Statement :-**

Diabetes is a prevalent chronic disease with significant implications for patient health and healthcare costs. Efficiently managing and predicting diabetes progression is crucial for timely intervention and personalized care. The challenge is to develop a robust MLOps (Machine Learning Operations) solution that leverages data-driven insights to enhance diabetes prediction models, deploy them in a scalable and automated manner, and continuously monitor and update the models to ensure accuracy and relevance. The goal is to establish an end-to-end MLOps framework that seamlessly integrates data processing, model development, deployment, and ongoing monitoring to optimize diabetes management and healthcare outcomes.

## **Objective :-**

Primary objective of this project are as follows :-

- Create a ML model for Diabetes data which can be found in sklearn and apply MLOPS using two ways.
  1. Azure ML
  2. Databricks
- Create a free Azure account.
- Using Azure Machine Learning Studio for creating ML pipeline which can be monitored and executed.
- We will also use Databricks for monitoring and executing our models.
- At last we will compare both ways.

# MLOPS using Azure Cloud

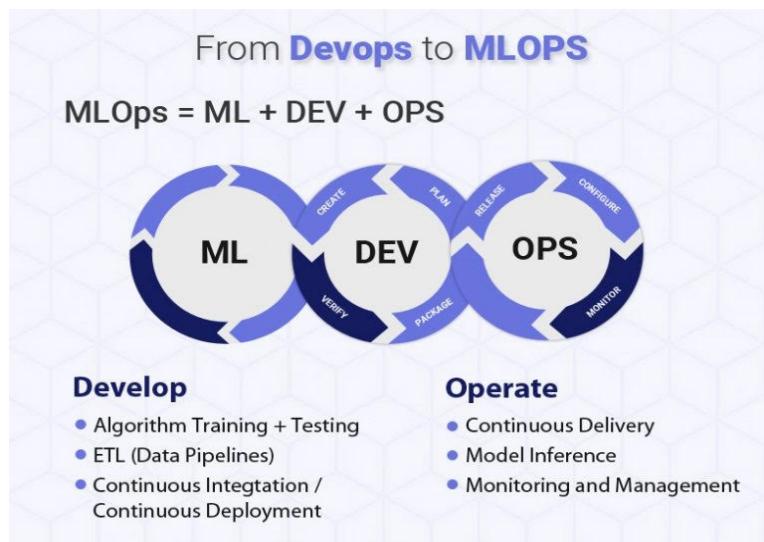
## Introduction

In our day-to-day life, we are doing automation in every domain with the help of Artificial Intelligence and Machine Learning. Now, building a Machine Learning Model requires so many steps which are as follows: -

(1) Defining the problem. (2) Data Preprocessing. (3) Exploratory Data Analysis. (4) Splitting the data. (5) Applying different Models/Algorithms. (6) Feature Scaling. (7) Train the Model. (8) Hyper-Parameter Tuning. (9) Evaluate the Model. (10) Deploy the Model. (11) Monitor and Maintain.

And if we want to make changes in the model so we must perform every step one by one again for deployment. So, we use MLOPS concept which will automatically build everything after we make any changes in the model.

We will see the basic introduction of MLOPS which is as follows: -



MLOPS, short for Machine Learning Operations, is a set of practices that aims to streamline and enhance the collaboration between data science and operations teams in the development and deployment of machine learning (ML) models. It is an emerging discipline that combines aspects of machine learning, software engineering, and IT operations to create a more efficient and scalable ML lifecycle.

## **Key Components of MLOps:**

### **1) Collaboration and Communication:**

MLOps emphasizes effective communication and collaboration between data scientists, engineers, and operations teams. This helps ensure that the entire ML pipeline, from model development to deployment and monitoring, is well-coordinated and aligned with business objectives.

### **2) Automation:**

Automation is a core principle of MLOps. It involves automating repetitive tasks throughout the ML lifecycle, such as data preprocessing, model training, testing, and deployment. Automation increases efficiency, reduces errors, and accelerates the overall development process.

### **3) Continuous Integration and Continuous Deployment (CI/CD):**

MLOps leverages CI/CD practices to enable the continuous integration of code changes, automated testing, and seamless deployment of ML models into production. CI/CD ensures that the ML pipeline is always in a deployable state, promoting a faster and more reliable release process.

### **4) Model Monitoring and Management:**

MLOps emphasizes the importance of monitoring ML models in production. This involves tracking model performance, detecting drift in data distribution, and addressing issues promptly. Additionally, MLOps provides tools for model versioning, enabling easy rollback and management of multiple model versions.

## **5)Scalability and Reproducibility:**

MLOps addresses the challenges of scaling ML workflows by providing infrastructure and tools that support the deployment of models in diverse environments. It also emphasizes reproducibility, ensuring that ML experiments and results can be replicated consistently.

## **6)Security and Compliance:**

Security is a critical consideration in MLOps, with a focus on safeguarding sensitive data and ensuring compliance with regulations. MLOps practices incorporate security measures throughout the ML lifecycle, from data handling to model deployment.

## **Why this is Project is Important:-**

- Early Detection and Prevention: Predictive models can assist in identifying individuals at risk of developing diabetes before the onset of symptoms. Early detection allows for proactive intervention and lifestyle modifications, potentially preventing or delaying the progression of the disease.
- Personalized Treatment Plans: Machine learning models can help tailor treatment plans for individuals based on their unique characteristics and health data. Personalized approaches may lead to more effective and efficient management of diabetes, optimizing medication regimens and lifestyle recommendations.

- Resource Allocation in Healthcare: By accurately predicting which individuals are more likely to develop complications or require intensive medical interventions, healthcare resources can be allocated more efficiently. This can help prioritize patient care and reduce the burden on healthcare systems.
- Public Health Planning: Aggregating and analyzing diabetes data on a larger scale can contribute to public health planning. Identifying trends, risk factors, and demographic patterns can inform public health policies, educational campaigns, and resource allocation at a broader level.
- Research and Understanding: Machine learning models applied to diabetes data can contribute to a deeper understanding of the underlying factors influencing the disease. This insight can guide further research, leading to the development of new treatments, medications, or preventive strategies.
- Continuous Monitoring and Feedback: In the context of wearable devices and remote patient monitoring, machine learning models can provide continuous feedback on a patient's health status. This real-time monitoring allows for timely adjustments to treatment plans and facilitates better disease management.
- Reducing Healthcare Costs: By preventing complications, optimizing treatment plans, and promoting proactive management, machine learning applications in diabetes care have the potential to reduce overall healthcare costs. This is particularly important given the economic burden associated with diabetes and its complications.

In summary, leveraging machine learning for diabetes data is important because it can lead to earlier intervention, personalized care, more efficient resource allocation, and a deeper understanding of the disease. These advancements contribute to improved patient outcomes, better public health strategies, and the overall well-being of individuals affected by diabetes.

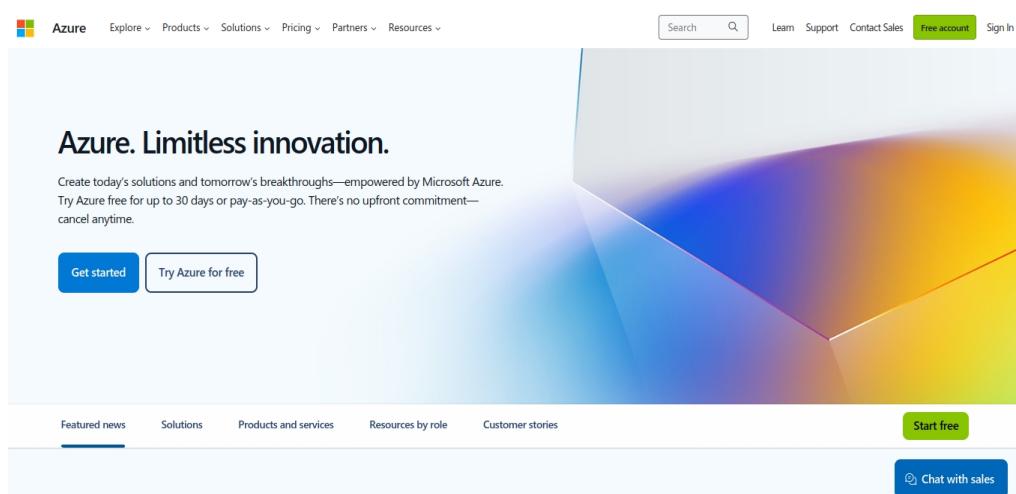
## **Why Azure ?**

We are using Azure Cloud because our system has limitation of Storage and RAM, and we want to make the model which requires huge memory and RAM to build that model and second reason is that Azure has better seamless Integration with other tools and Hybrid Cloud Solutions. To make this project first, we make an account on Azure Cloud, which provides basic services for free for one year and advance services for free for 30 days.

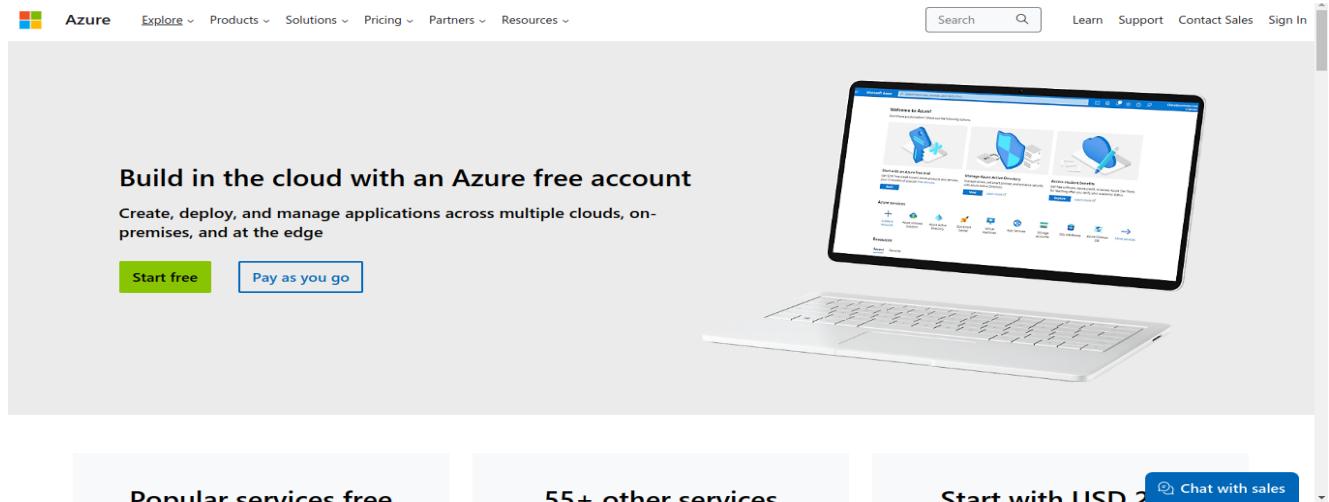
**So, there are so many tools which we can use in MLOPS but in this project we are trying to build it using a single tool, i.e., Azure Cloud and Databricks.**

**To create the Account on Microsoft Azure, following are the steps given below: -**

(1) Go to any web browser and type Microsoft Azure. Click on the first link. The window will appear whose screenshot is given below.



(2) Click on the green box Free Account on top Right Corner or Start Free on Bottom Right Corner. A new window will appear as follows: -



(3) Click on the Start Free Account. A New page will open which will ask for Sign Up. Put your Microsoft email id and get access to the free trial.

(4) Go to Microsoft Azure Portal. Your account will look like this in the below image.

A screenshot of the Microsoft Azure Portal dashboard. At the top, there's a header with 'Microsoft Azure', an 'Upgrade' button, a search bar ('Search resources, services, and docs (G+)'), and user profile information ('VipulAhuja@iitb.ac.in iIT-B (IITB.AC.IN)'). Below the header, there's a section titled 'Azure services' with icons for 'Create a resource', 'Subscriptions', 'Azure Cosmos DB', 'Microsoft Entra Connect', 'Microsoft Entra ID', 'Activity log', 'Azure AI Studio', 'All resources', 'Azure Machine Learning', and 'More services'. Underneath this, there's a 'Resources' section with tabs for 'Recent' (selected) and 'Favorite'. It lists recent resources: 'Vipul20.onmicrosoft.com' (B2C Tenant, 4 days ago), 'Free Trial' (Subscription, 4 days ago), 'mlopsytdemo3970472974' (Log Analytics workspace, 4 days ago), 'mlops\_yt\_demo' (Azure Machine Learning workspace, 4 days ago), 'mlops\_yt' (Resource group, 5 days ago), 'mlopsytdemo4835949336' (Application Insights, 6 days ago), and 'vipul' (Storage account, 6 days ago). At the bottom left, there's a link 'See all' and the URL 'https://portal.azure.com/#create/hub'.

## Now following are the steps for creating the Project: -

(1) Now search the resource group in the search bar at the top in the middle.

The screenshot shows the Microsoft Azure portal interface. At the top, there is a search bar with the text "resource groups". Below the search bar, there are several tabs: "All" (highlighted), "Services (31)", "Marketplace (1)", "Documentation (99+)", "Resources (0)", and "Resource Groups (0)". The main content area displays a list of items under the "Services" category, with "Resource groups" being the first item. Other items include "Subscriptions", "Keywords: resource groups", "Resource Guards", "Resource Graph Explorer", "Groups", "Application groups", and "Deleted Groups". On the left side, there is a sidebar with sections for "Azure services" (Create a resource, Subscriptions), "Recent" (Vipul20.onmicrosoft, Free Trial, milopsyt...), and "Name" (milopsyt...). At the bottom of the screenshot, the URL is shown as <https://portal.azure.com/#blade/HubsExtension/BrowseResourceGroupBlade/resourceType/Microsoft.Resources%2Fsubscriptions%2FresourceGroups>.

(2) Click on the resource group. Then New page will appear. On that click on Create.

The screenshot shows the Microsoft Azure "Resource groups" blade. At the top, there is a header with the title "Resource groups" and a "Create" button. Below the header, there are filter options: "Filter for any field...", "Subscription equals all", "Location equals all", and "Add filter". The main content area shows a single record: "Showing 1 to 1 of 1 records." The record details are: "Name" (milops\_yt), "Subscription" (Free Trial), and "Location" (West US 3). There are also sorting options: "Name ↑↓", "Subscription ↑↓", and "Location ↑↓". At the bottom right, there are buttons for "No grouping" and "List view".

(3) After that, a new Window will appear.

The screenshot shows the 'Create a resource group' page in the Microsoft Azure portal. The 'Basics' tab is selected. In the 'Project details' section, the 'Subscription' dropdown is set to 'Free Trial'. The 'Resource group' input field is empty. In the 'Resource details' section, the 'Region' dropdown is set to '(US) East US'. At the bottom, there are buttons for 'Review + create', '< Previous', and 'Next : Tags >'.

(4) In that set the Subscription Field as Free Trial. (By default, it is free trial). Free trial consists of \$200 worth of resources. Once it is used, your free trail expires, so whatever you use you should delete that resource so that you can maximize your usage. In resource Group, give the valid name of your choice. Select any region of your choice. I am selecting West US 3. Then Click on Review + create.

The screenshot shows the 'Create a resource group' page in the Microsoft Azure portal. The 'Basics' tab is selected. In the 'Project details' section, the 'Subscription' dropdown is set to 'Free Trial'. The 'Resource group' input field contains 'mt2022132'. In the 'Resource details' section, the 'Region' dropdown is set to '(US) West US 3'. At the bottom, there are buttons for 'Review + create', '< Previous', and 'Next : Tags >'.

(5) Now, Check if the validation pass comes or not. If it comes click on create, otherwise try again the above procedure.

Validation passed.

Basics Tags Review + create

Subscription: Free Trial  
Resource group: mt2022132  
Region: West US 3

Tags  
None

Create < Previous Next > Download a template for automation

(6) Goto Home. Click on Storage Account in the resource section.

Azure services

Storage accounts

Storage accounts

Recent Favorite

Name	Type	Last Viewed
mt2022132	Resource group	7 minutes ago
Vipul20.onmicrosoft.com	B2C Tenant	4 days ago
Free Trial	Subscription	4 days ago
mlopsytdemo3970472974	Log Analytics workspace	4 days ago
mlops_yt_demo	Azure Machine Learning workspace	4 days ago
mlops_yt	Resource group	6 days ago
mlopsytdemo4835949336	Application Insights	6 days ago
vipul	Storage account	6 days ago

(7) A new window will appear. Click on Create.

The screenshot shows the Microsoft Azure Storage accounts page. At the top, there's a search bar and a navigation bar with options like 'Home', 'Storage accounts', and user information ('Vipul.Ahuja@iitb.ac.in'). Below the navigation is a toolbar with buttons for '+ Create', 'Restore', 'Manage view', 'Refresh', 'Export to CSV', 'Open query', 'Assign tags', and 'Delete'. There are also filter and sorting options: 'Filter for any field...', 'Subscription equals all', 'Resource group equals all', 'Location equals all', and 'Add filter'. The main table displays one record:

Name	Type	Kind	Resource group	Location	Subscription
vipul	Storage account	StorageV2	mlops_yt	West US 3	Free Trial

At the bottom, there are pagination controls ('< Previous', 'Page 1 of 1', 'Next >') and a 'Give feedback' link.

(8) In the next step, set the subscription field as Free Trial. Set the Resource Group as the field name which you created in resource group in above steps. Here, I set it as “mt2022132”. So, I will select that. Set the valid storage name on your own. I am setting that as “mt2022132”. Click on next then again next till last next as rest field values are taken by default values.

The screenshot shows the 'Create a storage account' wizard in the 'Basics' step. The top navigation bar includes 'Microsoft Azure', 'Search resources, services, and docs', and user information ('VipulAhuja@iitb.ac.in'). Below the navigation is a breadcrumb trail: 'Home > Storage accounts > Create a storage account'. The main interface has tabs for 'Basics', 'Advanced', 'Networking', 'Data protection', 'Encryption', 'Tags', and 'Review'. The 'Basics' tab is selected. Under 'Project details', there are dropdown menus for 'Subscription' (set to 'Free Trial') and 'Resource group' (set to 'mlops\_yt'). Under 'Instance details', there is a dropdown menu for 'mt2022132'. At the bottom, there are buttons for 'Review' (highlighted in blue), '< Previous', 'Next : Advanced >', and a 'Give feedback' link.

(9) After setting default values of rest of the fields, you will see the image as follows

The screenshot shows the 'Create a storage account' review step in the Azure portal. The 'Review' tab is selected. The 'Basics' section displays the following configuration:

Subscription	Free Trial
Resource Group	mt2022132
Location	westus3
Storage account name	mt2022132
Deployment model	Resource manager
Performance	Standard
Replication	Read-access geo-redundant storage (RA-GRS)

The 'Advanced' section shows:

Enable hierarchical namespace	Disabled
Enable network file system v3	Disabled

At the bottom, there are buttons for 'Create', '< Previous' (disabled), 'Next >', 'Download a template for automation', and 'Give feedback'.

Click on Create now and wait for a few minutes as it takes some 3-5 minutes to create.

(10) As storage account has been created, click on Goto Resource button. You will see the details.

The screenshot shows the 'Deployment' overview page for the storage account 'mt2022132\_1706031933721'. The 'Overview' tab is selected. A message states 'Your deployment is complete'. Deployment details include:

- Deployment name: mt2022132\_1706031933721
- Subscription: Free Trial
- Resource group: mt2022132
- Start time: 23/01/2024, 23:15:46
- Correlation ID: b2d264cc-74e2-44fb-88a9-30e5a5a3a108

Next steps include 'Go to resource' and 'Give feedback'. A sidebar on the right provides links to 'Cost Management', 'Microsoft Defender for Cloud', 'Free Microsoft tutorials', and 'Work with an expert'.

**Essentials**

Resource group (move)	: mt2022132	Performance	: Standard
Location	: westus3	Replication	: Read-access geo-redundant storage (RA-GRS)
Primary/Secondary Location	: Primary: West US 3, Secondary: East US	Account kind	: StorageV2 (general purpose v2)
Subscription (move)	: Free Trial	Provisioning state	: Succeeded
Subscription ID	: 6a718482-f783-470a-8bb5-634ae459d6ba	Created	: 23/01/2024, 23:15:55
Disk state	: Primary: Available, Secondary: Available		

**Tags (edit)** : Add tags

**Properties**    Monitoring    Capabilities (7)    Recommendations (0)    Tutorials    Tools + SDKs

**Blob service**

Hierarchical namespace	Disabled	Require secure transfer for REST API operations	Enabled
Default access tier	Hot	Storage account key access	Enabled
Blob anonymous access	Enabled	Minimum TLS version	Version 1.2
Blob soft delete	Enabled (7 days)		

**Security**

(11) Now, go to the Home page. Search “Azure Machine Learning” in the search bar. Click on it. Then Click on Create button. Two options will appear. New Workspace and New Registry. Select 1<sup>st</sup> one that is New Workspace.

Name	Resource group	Type	Location	Subscription
mlops_yt_demo	mlops_yt	Azure Machine Learning workspace	West US 3	Free Trial

(12) Set the field as follows in the image.

The screenshot shows the 'Create a machine learning workspace' wizard. In the 'Resource details' section, 'Subscription' is set to 'Free Trial' and 'Resource group' is 'mt2022132'. In the 'Workspace details' section, 'Name' is 'mt2022132', 'Region' is 'West US 3', and 'Storage account' is 'mt2022132'. Buttons at the bottom include 'Review + create', '< Previous', 'Next : Networking', and 'Save'.

(13) The container registry is by default None. Click on Create below Content Registry dialog box to create new registry. Set the name as “mt2022132”. Set SKU as Standard. Click Save. Then click on Next: Networking.

The screenshot shows the 'Create a machine learning workspace' wizard. The validation status is 'Validation passed'. A 'Create new container registry' dialog box is open, showing 'Name' as 'mt2022132' and 'SKU' as 'Standard'. The main workspace creation form has 'Container registry' set to 'None'. Buttons at the bottom include 'Review + create', '< Previous', 'Next : Networking', 'Save', and 'Discard'.

(14) Click on Next and Next until last Next as rest field are set to be default value. In the end Validation passed will come. Then Click on create. Wait for a few minutes.

The screenshot shows the Microsoft MachineLearningServices | Overview page in the Azure portal. The deployment is marked as complete with a green checkmark. Deployment details include:

- Deployment name: Microsoft.MachineLearningServices
- Subscription: Free Trial
- Resource group: mt2022132
- Start time: 23/01/2024, 23:42:20
- Correlation ID: bdfc5945-5cd2-44d3-8389-864f3269...

Below the deployment details, there are sections for "Deployment details" and "Next steps". A prominent blue button labeled "Go to resource" is located at the bottom left of the main content area. To the right of the main content, there are several promotional cards:

- Cost management**: Get notified to stay within your budget and prevent unexpected charges on your bill. [Set up cost alerts >](#)
- Microsoft Defender for Cloud**: Secure your apps and infrastructure. [Go to Microsoft Defender for Cloud >](#)
- Free Microsoft tutorials**: Start learning today. [Start learning today >](#)
- Work with an expert**: Azure experts are service provider partners who can help manage your assets on Azure.

(15) Click on the Go to resource button.

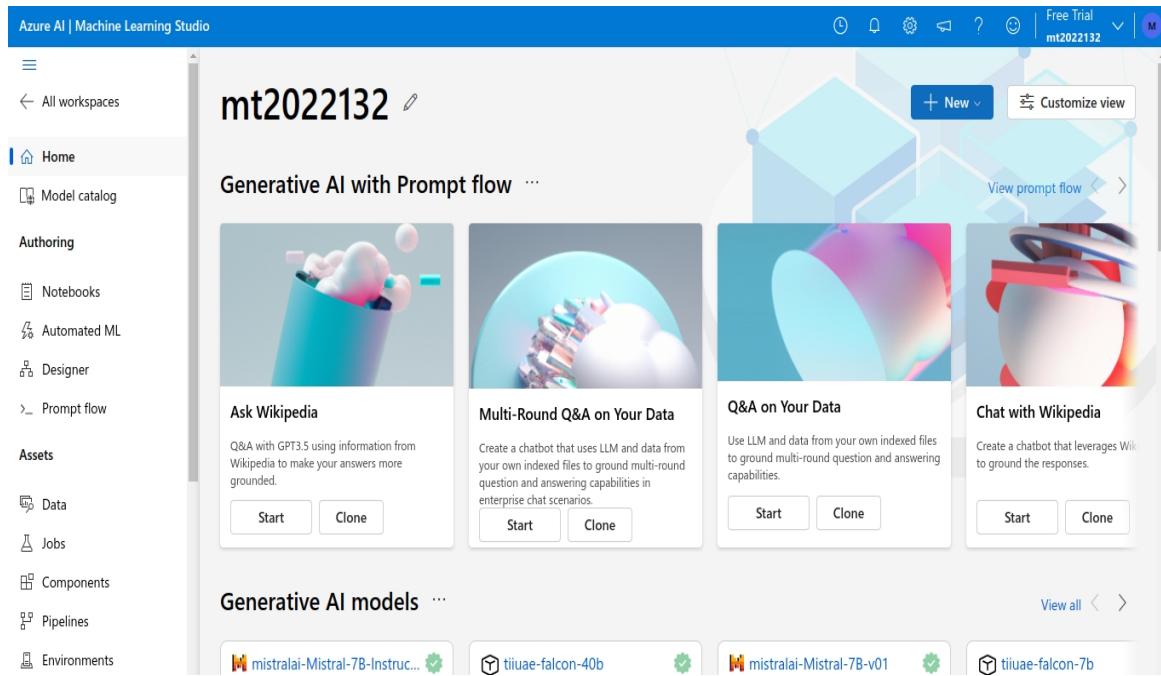
(16) A new window will appear. Click on the Studio Web URL link.

The screenshot shows the Azure Machine Learning workspace overview page for workspace mt2022132. The "Essentials" section displays the following information:

- Resource group: mt2022132
- Location: West US 3
- Subscription: Free Trial
- Storage: mt2022132
- Studio web URL: <https://ml.azure.com/?id=82a84c22-47b2-4612-b9f7-86...>
- Container Registry: mt2022132
- Key Vault: mt20221326697963552
- Application Insights: mt2022132910289020
- MLflow tracking URI: azurerm://westus3.api.azurerm.ms/mlflow/v1.0/subscripti...

At the bottom of the page, there is a call-to-action: "Work with your models in Azure Machine Learning Studio". It encourages users to start exploring or learn more about the Azure Machine Learning studio.

(17) It will Launch Azure AI/Machine Learning Studio in New Window. You can see that in the image below.



(18) In Studio, you will see so many options on the left panel which are divided into sections which can be seen on scrolling down. Using these options, we can integrate other MLOPS tools, or can create automated scripts, etc. But we are using only Azure Cloud for MLOPS so it has used very few options from that section because we are not using any different MLOPS tools (outside tools like Kubeflow, etc.)

(19) Now we will create an instance under Compute Section to assign some memory to our project. Go to Compute Section. Click on New.

Azure AI | Machine Learning Studio

iiit-b > mt2022132 > Compute

Compute

Designer

Prompt flow

Assets

- Data
- Jobs
- Components
- Pipelines
- Environments
- Models
- Endpoints

Manage

- Compute
- Monitoring PREVIEW
- Data Labeling
- Linked Services PREVIEW

Get started with Azure Machine Learning notebooks and R scripts by creating a compute instance

Choose from a selection of CPU or GPU instances preconfigured with popular tools such as VS Code, JupyterLab, Jupyter, and RStudio, ML packages, deep learning frameworks, and GPU drivers. [Learn more](#)

+ New

[View Azure Machine Learning tutorials](#)

[View available quota](#)

(20) Give the name of Compute. I am giving it as "mt2022132".

Create compute instance

Required settings

Configure required settings  
Select the name and virtual machine size you would like to use for your compute instance

Note that a compute instance can not be shared. It can only be used by a single assigned user. By default, it will be assigned to the creator and you can change this to a different user in the Security step.

Compute name \* mt2022132

Virtual machine type CPU

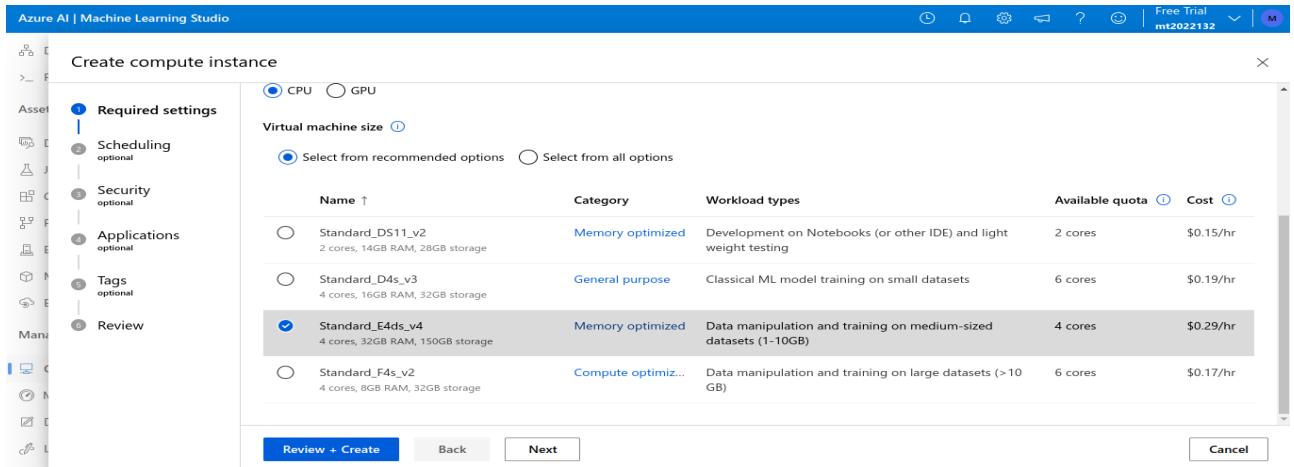
Virtual machine size Standard\_DS1\_v2

Select from recommended options  Select from all options

Name ↑	Category	Workload types	Available quota	Cost

Review + Create Back Next Cancel

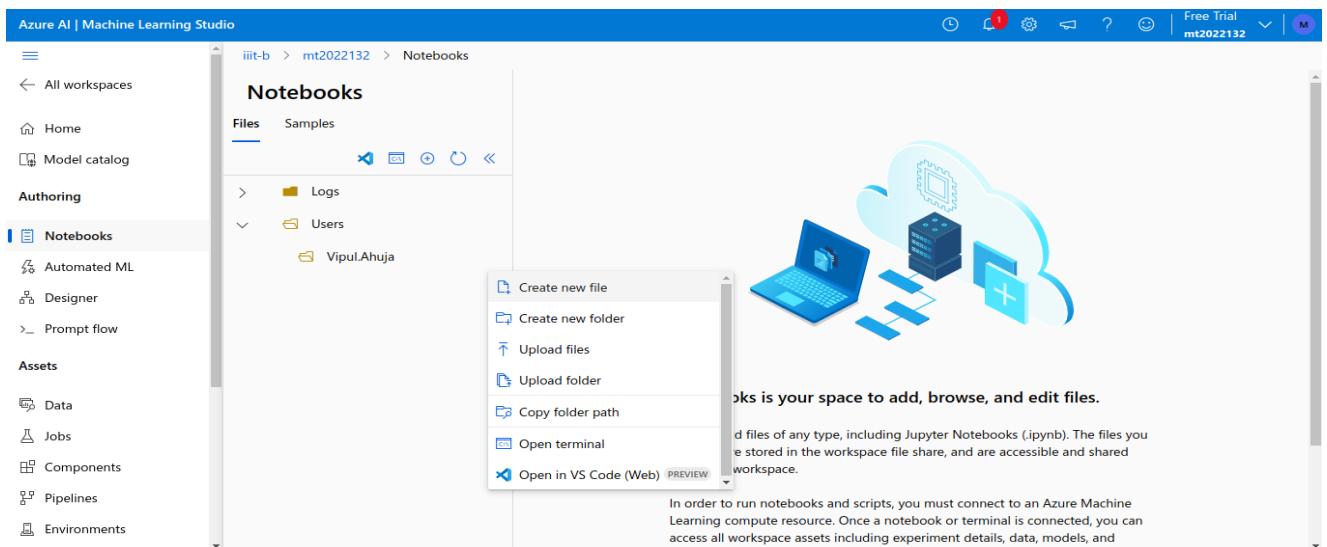
(21) Select the standard virtual machine size and delete another machine if you have created it, as it will cost you and you are not using it then you must delete that as creating 2 virtual machines instead of 1 will cost you double.



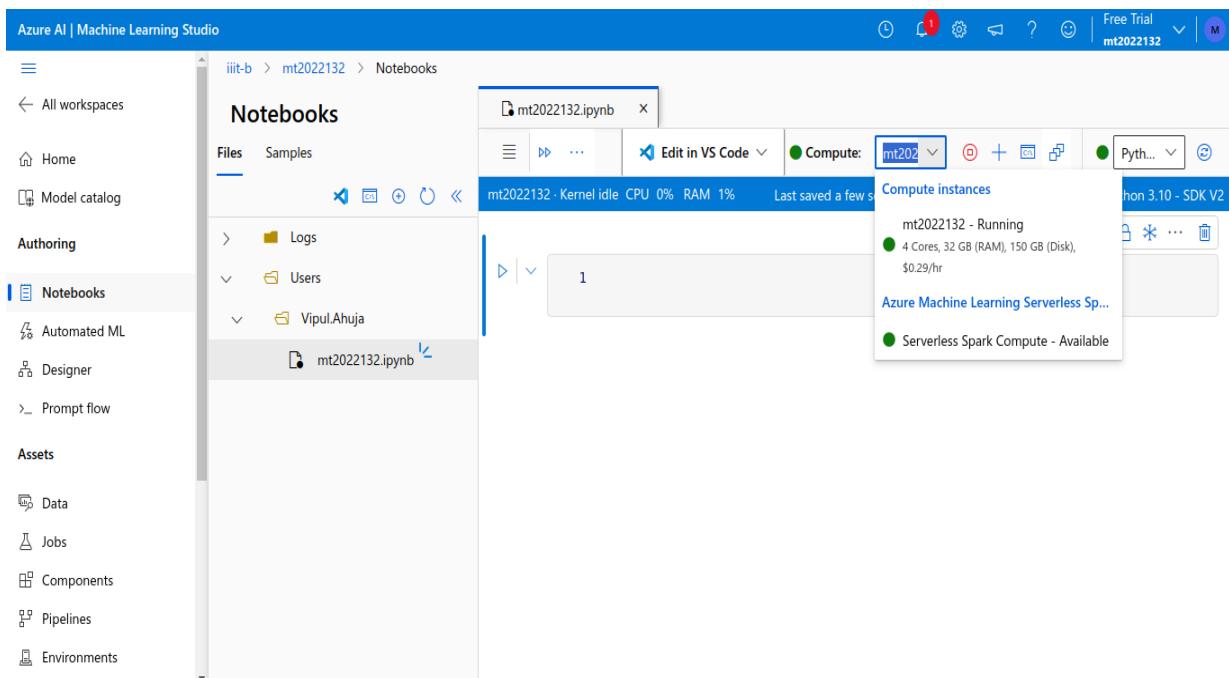
(22) Click on Next and set all the field values to be default. Then Click on Create. Wait for a few minutes. You can see the image below for the same.

Name	State	Idle shutdown	Applications	Size
mt2022132	Running	1 hour	JupyterLab Jupyter VS Code (Web) PREVIEW ...	STANDAR

(23) Now go to Notebook Section and Click on create new file under your Username folder as shown in the image. Then Click “Create new file” option. Create ipynb file.



(24) Write the valid name of the file on your own. I have set the name as “mt2022132”. Now, Select the compute instance that you have created. And make sure it is running and if it is not then start that.



(25) Now, write the code in ipynb file that you have created.

## Explanations :-

### MLOPS Using Azure ML Studio

Now we will use our diabetes data for Azure ML

Steps:-

- First we will import our data from sklearn

```
1 import joblib
2 import sklearn
3
4 from sklearn.datasets import load_diabetes
5 from sklearn.linear_model import BayesianRidge, Ridge
2] ✓
```

```
1 x, y = load_diabetes(return_X_y=True)
3] ✓
```

```
1 first_model = Ridge().fit(x, y)
2 joblib.dump(first_model, "model_ridge.pkl")
4] ✓
['model_ridge.pkl']
```

```
1 from azureml.core.model import Model
2 model = Model.register(model_path="model_ridge.pkl",
3                         model_name="model_ridge",
4                         workspace=ws)
5] ✓
```

- The score2.py script is designed for scoring with a Ridge regression model in Azure Machine Learning. The init() function initializes the model, loading it from the Azure Model Registry. The run() function handles inference by

parsing input JSON data, making predictions using the Ridge model, and returning results in JSON format. The script is intended for use in Azure ML scoring environments.

```
✓ 1  %%writefile score2.py
2  import joblib
3  import json
4  import numpy as np
5
6  from azureml.core.model import Model
7
8  def init():
9      global model_3
10     model_3_path = Model.get_model_path(model_name='model_ridge')
11     model_3 = joblib.load(model_3_path)
12
13 def run(raw_data):
14     try:
15         data = json.loads(raw_data)['data']
16         data = np.array(data)
17         result_1 = model_3.predict(data)
18
19         return {"prediction1": result_1.tolist()}
20     except Exception as e:
21         result = str(e)
22         return result
i] ✓
```

- This code deploys a machine learning model as an Azure Container Instances (ACI) web service using Azure Machine Learning. It configures a Python environment, specifies an inference script (score2.py), and deploys the model to ACI with defined CPU and memory resources. The code waits for the deployment to complete and prints the service state, providing an end-to-end demonstration of model deployment in Azure ML.

```
... LX VT L
1  from azureml.core.webservice import AciWebservice
2
3  aci_service_name = "aciservice-modelridge"
4
5  deployment_config = AciWebservice.deploy_configuration(cpu_cores=1, memory_gb=1)
6
7  service = Model.deploy(ws, aci_service_name, [model], inference_config, deployment_config, overwrite=True)
8  service.wait_for_deployment(True)
9
10 print(service.state)
✓
```

- At last we will get our results

```
1 import json
2 test_sample = json.dumps({'data': x[0:2].tolist()})
3 predictions = service.run(test_sample)
4 predictions
✓
{'prediction1': [182.67335420683418, 90.99860655841785]}
```

Now we will use of Model tab in Azure ML studio which helps in following ways

- Model Registration:
  - Register trained models in the "Models" section.
  - Create model references, store metadata, and enable versioning.
- Model Deployment:
  - Deploy registered models as web services.
  - Deployment options include Azure Kubernetes Service (AKS) and Azure Container Instances (ACI).
- Endpoint Management:
  - Manage model endpoints for monitoring, scaling, and updates.
  - Ensure deployed models meet performance requirements.
- Model Versioning:
  - Utilize versioning to track different model iterations.
  - Facilitates maintaining a history of model changes and enables rollback.
- Experimentation and Run History:
  - Access experiment tracking and run history features.
  - Monitor performance and results of various model training runs.
- Pipeline Integration:

- o Integrate model deployment into end-to-end ML pipelines.
- o Automate the entire ML lifecycle, from data preparation to deployment.
- Comprehensive Platform:
  - o Azure Machine Learning Studio serves as a central platform for model management.
  - o Offers a range of features for registration, deployment, versioning, and integration.

The screenshot shows the 'Details' tab of a model named 'model\_ridge:1'. The top navigation bar includes links for 'iiit-b', 'mllopsML', 'Models', and 'model\_ridge:1'. Below the navigation are buttons for 'Refresh', 'Archive', 'Deploy', 'Download all', and 'Share model'. The main content area is divided into sections: 'Attributes', 'Tags', 'Properties', and 'Description'. The 'Attributes' section contains fields for Name (model\_ridge), Version (1), Created on (Jan 23, 2024 3:54 PM), Created by (MT2022048 Harsh Tripathi), Type (CUSTOM), and Asset ID (azurerm://locations/centralindia/workspaces/403dad89-a867-4666-83d4-6927da701487/models/model\_ridge/versions/1). The 'Tags' section indicates 'No tags'. The 'Properties' section indicates 'No properties'. The 'Description' section has a placeholder 'Click edit icon to add a description'.

We can also use endpoints to deploy the model.

The screenshot shows the Azure ML studio interface. The top navigation bar includes 'mlopsML', 'Models', 'model\_ridge:1', 'Endpoints', and 'aciservice-modelridge'. The main title is 'aciservice-modelridge'. Below the title, there are tabs for 'Details', 'Test', 'Consume', and 'Logs', with 'Details' being the active tab. The 'Details' section contains various configuration parameters:

Deployment state	Healthy
Operation state	Succeeded
Compute type	Container instance
Created by	MT2022048 Harsh Tripathi
Model ID	model_ridge:1
Created on	Jan 23, 2024 3:54 PM
Last updated on	Jan 23, 2024 3:54 PM
Image ID	--
REST endpoint	<a href="http://9cc3dc00-694f-4de0-8204-a8bfa31be6f8.centralindia.azurecontainer.io/score">http://9cc3dc00-694f-4de0-8204-a8bfa31be6f8.centralindia.azurecontainer.io/score</a>
Key-based authentication enabled	false
Swagger URI	<a href="http://9cc3dc00-694f-4de0-8204-a8bfa31be6f8.centralindia.azurecontainerio/swagger.json">http://9cc3dc00-694f-4de0-8204-a8bfa31be6f8.centralindia.azurecontainerio/swagger.json</a>
CPU	1
Memory	1 GB

On the right side of the interface, there is a panel with three configuration settings:

hasInferenceSchema	True
hasHttps	False
authEnabled	False

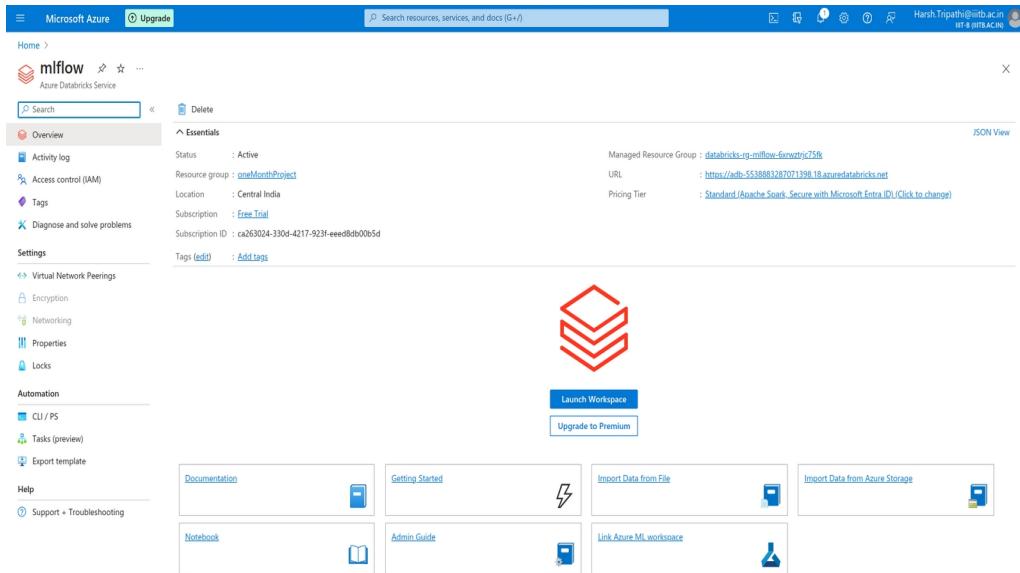
So by this way we can deploy our model to real world and feed live data.

## We Will also implement the project in Databricks

Databricks is a cloud-based platform designed for big data analytics and machine learning. It was founded by the creators of Apache Spark, a widely used open-source distributed computing system. Databricks provides an integrated environment that facilitates data engineering, data science, and collaborative data analytics.

Steps for creating databricks workspace in azure:-

- In the Azure Portal, click on "Create a resource."
- Search for "Databricks" in the Azure Marketplace and select "Azure Databricks."
- Click "Create" to start the creation process.
- Fill in the required information, such as subscription, resource group, workspace name, region, and pricing tier.
- Configure additional settings and click "Review + create."
- Review your configuration and click "Create" to deploy the Databricks workspace.



At last, launch the workspace to open Databricks. We need to create a new notebook for writing our codes, for using mlops we will MLFlow.

MLflow is an open-source platform designed to manage the end-to-end machine learning lifecycle. Developed by Databricks, MLflow provides tools for tracking experiments, packaging code into reproducible runs, and sharing and deploying models.

MLflow supports various machine learning libraries and frameworks, making it agnostic to the underlying technology. It has integrations with popular tools like scikit-learn, TensorFlow, and PyTorch. MLflow is cloud-agnostic and can be used both on-premises and in cloud environments.

Overall, MLflow enhances collaboration and reproducibility in machine learning projects, addressing challenges across the entire lifecycle, from experimentation and development to deployment and monitoring. It has gained popularity in the machine learning community as a versatile and user-friendly tool for managing the complexities of machine learning workflows.

## Steps:-

- Analysis of data

```
CMD /  
  
1 Y=np.array([y]).transpose()  
2 d=np.concatenate((X,Y),axis=1)  
3  
4 columns = ['age', 'sex', 'bmi', 'bp', 's1', 's2', 's3', 's4', 's5', 's6', 'progression']  
5 data=pd.DataFrame(d,columns=columns)  
  
Command took 0.10 seconds -- by harsh.tripathi@iitb.ac.in at 1/28/2024, 11:40:51 PM on MT2022048 Tripathi's Cluster  
  
Cnd 8  
  
1 data.head()  
  


|   | age       | sex       | bmi       | bp        | s1        | s2        | s3        | s4        | s5        | s6        | progression |
|---|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-------------|
| 0 | 0.038076  | 0.050680  | 0.061696  | 0.021872  | -0.044223 | -0.034821 | -0.043401 | -0.002592 | 0.019907  | -0.017646 | 151.0       |
| 1 | -0.001882 | -0.044642 | -0.051474 | -0.026328 | -0.008449 | -0.019163 | 0.074412  | -0.039493 | -0.068332 | -0.092204 | 75.0        |
| 2 | 0.085299  | 0.050680  | 0.044451  | -0.005670 | -0.045599 | -0.034194 | -0.032356 | -0.002592 | 0.002861  | -0.025930 | 141.0       |
| 3 | -0.089063 | -0.044642 | -0.011595 | -0.036656 | 0.012191  | 0.024991  | -0.036038 | 0.034309  | 0.022688  | -0.009362 | 206.0       |
| 4 | 0.005383  | -0.044642 | -0.036385 | 0.021872  | 0.003935  | 0.015596  | 0.008142  | -0.002592 | -0.031988 | -0.046641 | 135.0       |

  
Command took 0.10 seconds -- by harsh.tripathi@iitb.ac.in at 1/28/2024, 11:40:51 PM on MT2022048 Tripathi's Cluster
```

- The plot\_elastic function generates and saves a plot illustrating the regularization path of coefficients for an Elastic Net model. It uses the enet\_path function to calculate the model coefficients for different regularization strengths. The resulting plot, showing how coefficients change with varying regularization, is saved as 'Elasticnet\_path.png'.

```

def plot_elastic(X,y,l1_ratio):
    eps=5e-3
    alpha_enet, coef_enet, _ =enet_path(X , y, eps=eps, l1_ratio=l1_ratio)
    global image

    fig= plt.figure()
    ax=plt.gca()

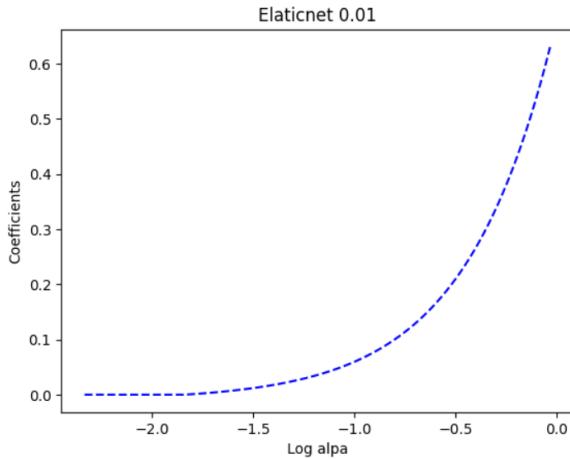
    colors = cycle(['b','r','g','c','k'])
    neg_alpha_ents = -np.log10(alpha_enet)

    for coef_e , c in zip(coef_enet, colors):
        l1=plt.plot(neg_alpha_ents, coef_e, c=c, linestyle = '--')
        plt.xlabel('Log alpa')
        plt.ylabel('Coefficients')

    title="Elasticnet "+ str(l1_ratio)
    plt.title(title)
    plt.axis('tight')
    image=fig
    fig.savefig('Elasticnet_path.png')
    plt.close(fig)
    return image

```

- The `train_diab_model` function trains an Elastic Net regression model on diabetes data, evaluates its performance, logs the results using MLflow, and saves the trained model. It splits the data, handles hyperparameters, trains the model, logs hyperparameters and metrics, saves the model and its artifacts, and returns information about the MLflow run.



- Once our model is ready, we will print some info, The `print_model_info` function prints details about each model version. The subsequent code uses the MLflow tracking client to create a registered model and a new model version associated with that model in the MLflow model registry. If the model already exists, it catches the exception. A delay is added to allow time for the model version to be fully registered.
- The code retrieves information about the latest version of a registered model, prints the details (though not captured due to the function not returning anything), and transitions the staging stage of that model version to "staging" in the MLflow model registry.

```
1 model_info = print_model_info(client.get_latest_versions(name=model_name))

name diab
version 3
run id a39902d944624f56a72317791d8d0a66
current_stage Staging
name diab
version 4
run id 50036fb3ba794a8b9a095fe321be89e4
current_stage None

Command took 0.20 seconds -- by harsh.tripathi@iiitb.ac.in at 1/28/2024, 11:40:51 PM on MT2022048 Tripathi's Cluster
```

Cmd 19

```
1 client.transition_model_version_stage(model_name, model_version.version, stage="staging")

<ModelVersion: aliases=[], creation_timestamp=1706465473907, current_stage='Staging', description='', e321be89e4', run_link='', source='dbfs:/databricks/mlflow-tracking/2389151175013304/50036fb3ba794a8b9_id='4490419882523069', version='4'>

Command took 0.29 seconds -- by harsh.tripathi@iiitb.ac.in at 1/28/2024, 11:40:51 PM on MT2022048 Tripathi's Cluster
```

## Details about the Experiment:-

- The "Experiment" tab in Databricks is part of the MLflow integration.
- Users can create different experiments to organize and track machine learning runs.
- It allows logging runs with parameters, metrics, and artifacts for each machine learning task.
- Users can compare runs within the same experiment based on metrics, parameters, or metadata.
- The tab provides visualization of metrics over time, aiding in understanding model performance variations.
- Artifacts, such as model files, are automatically logged and can be managed within the "Experiment" tab.
- Runs within experiments are versioned, enabling tracking of changes over time.
- Experiments can be shared with collaborators for collaborative machine learning development.

- The tab supports the entire machine learning lifecycle, including experimentation, model packaging, deployment, and monitoring.

Experiments > [mlflow\\_diab](#)

Experiment ID: 2389151175013304 Artifact Location: dbfs/databricks/mlflow-tracking/2389151175013304

> Description [Edit](#)

		Run Name	Created	Dataset	Duration	Source	Models
<input type="checkbox"/>		bold-crab-680	12 minutes ago	-	6.2s	mlflow_...  diab/4	
<input type="checkbox"/>		colorful-finch-549	12 minutes ago	-	6.6s	mlflow_...  sklearn	
<input type="checkbox"/>		skillful-mouse-648	12 minutes ago	-	4.8s	mlflow_...  sklearn	
<input type="checkbox"/>		clean-donkey-197	36 minutes ago	-	6.4s	mlflow_...  diab/3	
<input type="checkbox"/>		agreeable-midge-458	37 minutes ago	-	7.9s	mlflow_...  sklearn	

Experiments > [/Users/harsh.tripathi@iiitb.ac.in/mlflow\\_diab](#) > **bold-crab-680**

[Overview](#) [Model metrics](#) [System metrics](#) [Artifacts](#)

Run ID	50036fb3ba794a8b9a095fe321be89e4
Duration	6.2s
Datasets used	—
Tags	<a href="#">Add</a>
Source	mlflow_diab
Logged models	sklearn
Registered models	diab v4

**Parameters (5)**

Search parameters	
Parameter	Value
RMSE	78.41040260732004
alpha	4.65
I1 ration	0.21
mae	64.92732164792803
r2	-0.0026694073382835803

**Metrics (0)**

--

In this we can go to Artifacts tab to see further details of our model like yaml files:-

The screenshot shows the Databricks UI for a registered MLflow model named "bold-crab-680". The "Artifacts" tab is selected. On the left, there is a file tree under the "model" directory containing "MLmodel", "conda.yaml", "model.pkl", "python\_env.yaml", and "requirements.txt". To the right, the "MLflow Model" section displays code snippets for "Model schema" and "Make Predictions". The "Model schema" section indicates "No schema. See MLflow docs for how to include input and output schema with your model." The "Make Predictions" section provides examples for "Predict on a Spark DataFrame" and "Predict on a Pandas DataFrame".

```
Full Path:dbfs:/databricks/mlflow-tracking/2389151175013304/50036fb3ba794a8b9a095fe321be89e4/artifacts/model  
diab.v4  
Registered on 2024/01/28
```

**MLflow Model**  
The code snippets below demonstrate how to make predictions using the logged model. This model is also registered to the [model registry](#).

**Model schema**  
Input and output schema for your model. [Learn more](#)

Name	Type
No schema. See <a href="#">MLflow docs</a> for how to include input and output schema with your model.	

**Make Predictions**  
Predict on a Spark DataFrame:

```
import mlflow
from pyspark.sql.functions import struct, col
logged_model = 'runs:/50036fb3ba794a8b9a095fe321be89e4/model'

# Load model as a Spark UDF. Override result_type if the model does not
# return double values.
loaded_model = mlflow.pyfunc.spark_udf(spark, model_uri=logged_model,
                                         result_type='double')

# Predict on a Spark DataFrame.
df.withColumn('predictions', loaded_model(struct(*map(col, df.columns))))
```

Predict on a Pandas DataFrame:

```
import mlflow
logged_model = 'runs:/50036fb3ba794a8b9a095fe321be89e4/model'

# Load model as a PyFuncModel.
loaded_model = mlflow.pyfunc.load_model(logged_model)
```

The screenshot shows the Databricks UI for a registered MLflow model named "bold-crab-680". The "Artifacts" tab is selected. On the left, there is a file tree under the "model" directory containing "MLmodel", "conda.yaml", "model.pkl", "python\_env.yaml", and "requirements.txt". To the right, the "MLflow Model" section displays detailed configuration parameters for the model. These include "artifact\_path: model", "databricks\_runtime: 14.2.x-cpu-ml-scala2.12", "flavors: python\_function", "env: conda: conda.yaml, virtualenv: python\_env.yaml, loader\_module: mlflow.sklearn, model\_path: model.pkl, predict\_fn: predict, python\_version: 3.10.12", "sklearn: code: null, pickled\_model: model.pkl, serialization\_format:云pickle, sklearn\_version: 1.1.1", "mlflow\_version: 2.8.1", "model\_size\_bytes: 769", "model\_uuid: 76a68a6a53ac45b0afa8dc6e8aa9d49", "run\_id: 50036fb3ba794a8b9a095fe321be89e4", and "utc\_time\_created: '2024-01-28 18:11:07.641322'".

```
Full Path:dbfs:/databricks/mlflow-tracking/2389151175013304/50036fb3ba794a8b9a095fe321be89e4/artifacts/model/...  
Size: 569B
```

**MLflow Model**  
Full Path:dbfs:/databricks/mlflow-tracking/2389151175013304/50036fb3ba794a8b9a095fe321be89e4/artifacts/model/  
artifact\_path: model  
databricks\_runtime: 14.2.x-cpu-ml-scala2.12  
flavors:  
python\_function:  
env:  
conda: conda.yaml  
virtualenv: python\_env.yaml  
loader\_module: mlflow.sklearn  
model\_path: model.pkl  
predict\_fn: predict  
python\_version: 3.10.12  
sklearn:  
code: null  
pickled\_model: model.pkl  
serialization\_format: cloudpickle  
sklearn\_version: 1.1.1  
mlflow\_version: 2.8.1  
model\_size\_bytes: 769  
model\_uuid: 76a68a6a53ac45b0afa8dc6e8aa9d49  
run\_id: 50036fb3ba794a8b9a095fe321be89e4  
utc\_time\_created: '2024-01-28 18:11:07.641322'

Now we will see Models tab in Databrick,

- Model Registration: The "Model" tab in Databricks, often associated with MLflow's Model Registry, allows users to register and organize machine learning models.
- Version Control: Users can manage different versions of a model, enabling version control, comparisons, and rollbacks to maintain consistency and reliability.
- Approval Workflow: The Model Registry supports approval workflows, where models can progress through stages like "Staging" and "Production" with appropriate approvals.
- Deployment: Registered models can be easily deployed from the Model Registry to production environments for serving predictions.
- Access Control: Access controls are implemented to ensure that only authorized users can modify or deploy models, enhancing security and governance.
- Monitoring and Logging: The Model Registry provides capabilities for monitoring model performance and logging relevant metrics during the model lifecycle.
- Collaboration: Facilitates collaboration among data scientists and stakeholders by serving as a central hub for managing and tracking the lifecycle of machine learning models.

Registered Models >  
diab

[Details](#) Legacy serving

Notify me about [All new activity](#)

Created Time: 2024-01-23 14:56:10 Last Modified: 2024-01-28 23:41:23 Creator: harsh.tripathi@iiitb.ac.in

> Description [Edit](#)

> Tags

> Versions [All](#) Active 3 [Compare](#)

Version	Registered at	Created by	Stage	Pending Requests	Description	Endpoints
Version 4	2024-01-28 23:41:13	harsh.tripathi@iiitb...	Staging	-	-	-
Version 3	2024-01-28 23:16:29	harsh.tripathi@iiitb...	Staging	-	-	-
Version 2	2024-01-23 15:01:19	harsh.tripathi@iiitb...	Staging	-	-	-
Version 1	2024-01-23 14:58:52	harsh.tripathi@iiitb...	None	-	-	-

1

As per our model we can change the states of it,

The screenshot shows the MLflow UI for a model version named 'Version 4'. At the top right, there is a blue button labeled 'Use model for inference'. Below the header, there are sections for 'Request' (Request by: harsh.tripathi@iiitb.ac.in), 'Follow Status' (Following), 'Source Run' (bold-crab-680), and 'Stage' (Staging). A dropdown menu for 'Stage' is open, showing 'Staging' as the current stage. There are also sections for 'Description' (Edit), 'Pending Requests', 'Tags', 'Schema', and 'Activities'. An activity log entry shows a stage transition from 'None' to 'Staging' 22 minutes ago by 'harsh.tripathi@iiitb.ac.in'.

We can also use our model for inference such that we can create endpoint so that we can feed data from other sources.

The dialog box is titled 'Set up model inference' and has a close button 'X'. It contains instructions: 'Select one of real-time inference, streaming via Delta Live Tables, or batch.' Below this, there are three tabs: 'Real-time' (underlined), 'Streaming (Delta Live Tables)', and 'Batch inference'. The 'Real-time' tab is selected. The form fields include:

- \* Model: diab
- \* Model version: Version 4
- \* Endpoint name: diab
- Compute:
  - Small
  - 0-4 concurrency (0-4 DBU)
  - Scale to zero

At the bottom are 'Cancel' and 'Create endpoint' buttons.

## **Conclusion :-**

### **A comparision between the two ways to apply MLOPS :-**

<b>Feature</b>	<b>Azure ML Studio</b>	<b>Databricks</b>
<b>Purpose</b>	Comprehensive ML platform for model development and deployment.	Unified analytics platform for big data processing, analytics, and collaborative data science.
<b>Model Building</b>	Drag-and-drop interface for building ML models. Supports automated machine learning (AutoML).	Supports machine learning using MLlib and other popular ML libraries.
<b>Integration with Azure</b>	Tightly integrated with Azure services for data storage, integration, and deployment.	Can be integrated with Azure, providing access to Azure Data Lake Storage, Azure Blob Storage, and other services.
<b>Data Preparation</b>	Provides data preparation tools within the studio.	Offers interactive notebooks for data exploration and analysis.
<b>Deployment Options</b>	Supports deployment of models as web services (e.g., Azure Kubernetes Service, Azure Container Instances).	Focuses on big data processing and analytics rather than direct model deployment.
<b>Collaboration</b>	Suitable for individual or team-based ML model development.	Facilitates collaboration among data scientists, data engineers, and analysts in a shared workspace.
<b>Experimentation and Tracking</b>	Allows tracking and logging of experiments	Provides collaborative notebook development

	for understanding model performance.	with tracking features.
<b>Scalability</b>	Well-suited for smaller to medium-scale ML tasks.	Highly scalable, optimized for distributed data processing using Apache Spark.
<b>Big Data Processing</b>	Less emphasis on big data processing.	Primary focus on big data processing, leveraging Apache Spark.
<b>Use Case</b>	Ideal for end-to-end ML workflows.	Ideal for big data analytics, data engineering, and collaborative data science.

In summary, Azure ML Studio is tailored for end-to-end machine learning workflows, while Databricks is a unified analytics platform with a strong emphasis on big data processing and collaborative data science. The choice between them depends on the specific requirements of the project and the desired focus, whether it's on machine learning or big data analytics.

At the End it's use choice.