Machine Learning Model Deployment with IBM Cloud Watson Studio

Abstract:
Machine learning model deployment is a critical step in turning data science projects into practical, real-world solutions. IBM Cloud Watson Studio provides a comprehensive platform for deploying machine learning models, offering a seamless integration of data preparation, model development, and deployment. In this document, we present a comprehensive guide to deploying machine learning models using IBM Cloud Watson Studio, encompassing key concepts, best practices, and step-by-step instructions. Whether you are a data scientist looking to deploy your models or an IT professional responsible for managing the deployment process, this guide will equip you with the knowledge and tools necessary for successful model deployment.

This comprehensive guide will provide a step-by-step approach to deploying machine learning models using IBM Cloud Watson Studio, ensuring that your models are not only deployed successfully but also managed efficiently and securely in a production environment. Whether you are new to model deployment or looking to enhance your existing deployment practices, this guide will serve as a valuable resource.

Implementation

Step-1: Login to IBM Cloud here.

Step -2: Go to catalog and create a Watson Studio service in AI category.

Step-3: Click on Get started to launch Watson Studio Dashboard

Step-4: Create a project in IBM Watson Studio Dashboard and assign a Cloud object Storage service to manage datasets

Note: Cloud Object Storage is a storage service in IBM Cloud. We use this service to manage our datasets for training the ML Model and store required files. For more info check out the documentation here

Step-5: Add a jupyter notebook instance in your project to Develop and Deploy Machine Learning Model.

Note: You can either create a blank notebook or import from existing file or URL.

Step-6: Build a Machine Learning model using jupyter notebook instance.

Sample program;

```
Import numpy as np
Import matplotlib.pyplot as plt
Import pandas as pd
#Check Missing Values
Dataset.isnull().any()
#Spilt Dependent and Independent Variables
X = dataset.iloc[:, [2, 3]].values
Y = dataset.iloc[:, 4].values
# Splitting the dataset into the Training set and Test set
From sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=
0.25, random_state = 0)
# Fitting Decision Tree Classification to the Training set
From sklearn.tree import DecisionTreeClassifier
Classifier = DecisionTreeClassifier(criterion = 'entropy', random_state =
0)
Classifier.fit(X_train, y_train)
# Predicting the Test set results
Y_pred = classifier.predict(X_test)
#Finding the accuracy score
From sklearn.metrics import accuracy_score
Print("Accuracy Score: ",accuracy_score(y_test,y_pred)*100,"%")
```

Now we can proceed saving and deploying our Machine Learning model in Watson Machine Learning Repository.

So, before writing a code for deploying a model let's create a Watson Machine Learning Service.

Step 7: Go back to catalog and create Machine Learning service in AI Category

Step 8: Go to service credentials, create a new credential and copy them for further use.

Step -9: Import WatsonMachineLearningAPIClient library. Watson studio uses Watson Machine Learning service credentials to access WML service, so paste the credentials.

```
From watson_machine_learning_client import
WatsonMachineLearningAPIClient
Wml_credentials={
  "url": "xxxxxxxxxxxxxxxxxx",
  “apikey”: “xxxxxxxxxxxxxxxxxxxxxxx”,
  “username”: “xxxxxxxxxxxxxxxxxxxx”,
  “password”: “xxxxxxxxxxxxxxxxxxxxxxx”,
  “instance_id”: “xxxxxxxxxxxxxxxxxxxxxx”
}
```
Initialize the WML API Client

```
Client = WatsonMachineLearningAPIClient(wml_credentials)
```
Step-10: In this step we have to specify our machine learning model properties and store the model in WML repository.

Note: Here classifier is our Machine Learning model instance

```
#Specify the Properties
Model_props = {client.repository.ModelMetaNames.AUTHOR_NAME: “Abhi”,
               Client.repository.ModelMetaNames.AUTHOR_EMAIL: “”,
               Client.repository.ModelMetaNames.NAME: “MyModel”}
#Store the Machine Learning Model
Model_artifact=client.repository.store_model(classifier,
meta_props=model_props)
```
You can find the list of saved models

```
Client.repository.list()
```

Step- 11: Now we are ready to deploy our machine learning model as a Web service

```
#Get model UID
Published_model_uid = client.repository.get_model_uid(model_artifact)
#Deploy the model
Created_deployment = client.deployments.create(published_model_uid,
name=”MyDeployment”)
```

Deploy Success ☺
Finally we are done deploying our machine learning model.

Here is the code to get the scoring endpoint of our deployed Machine learning model.

```
Scoring_endpoint = client.deployments.get_scoring_url(created_deployment)
Print(scoring_endpoint)
```
Here is the sample code to test the scoring endpoint and get predictions

```
Scoring_payload = {“fields”: [“Age”,”Salary”],”values”:[[25,50000]]}
Predictions = client.deployments.score(scoring_endpoint, scoring_payload)
Print(predictions)
```