Phase-4
Development part - 2
Project:Machine Learning Model Deployment with IBM WatsonStudio (ISP Churn Model)

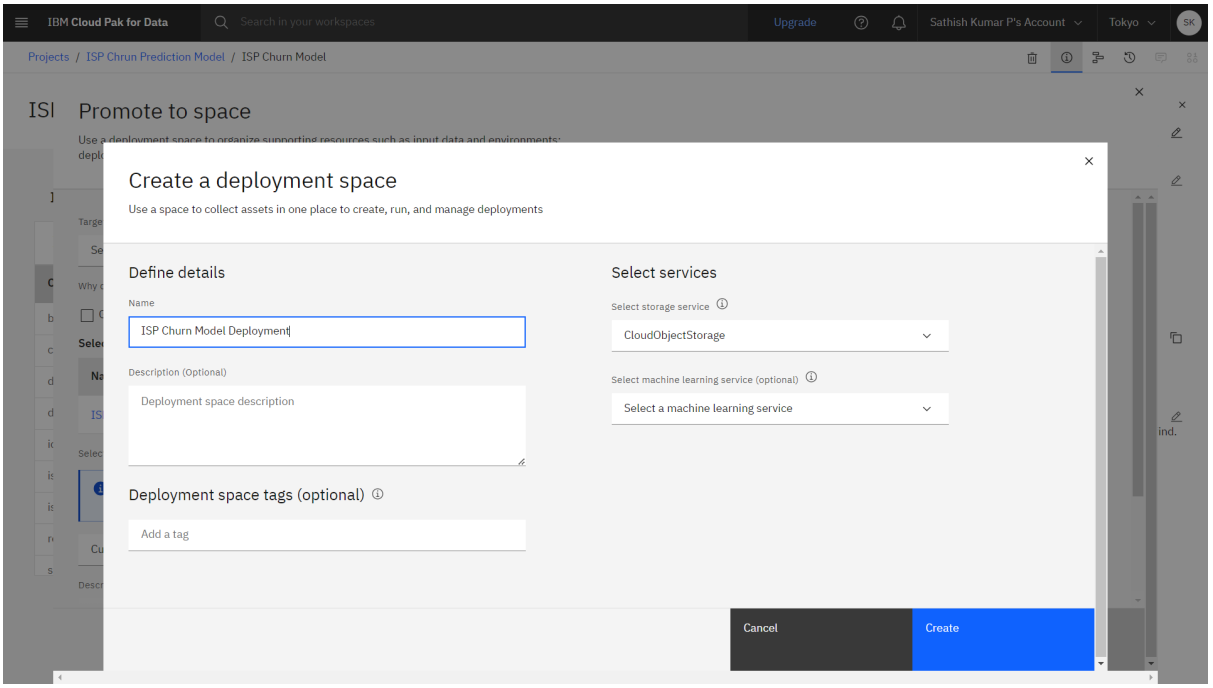
Introduction:

As far we created a model ,then the next step is to deploy the model and integrate the model to web service.The below steps are we done to deploy the model and integrate the model to web service.

Model Deployment:

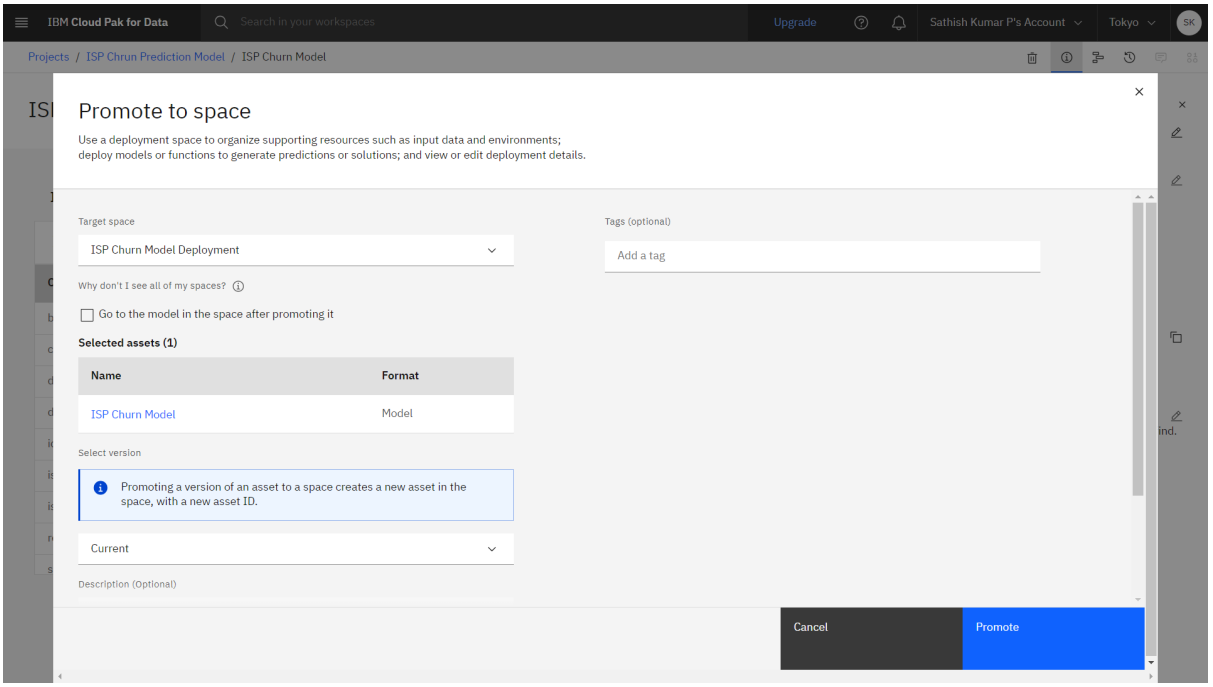
Step:01

We need to create a deployment space to deploy the model



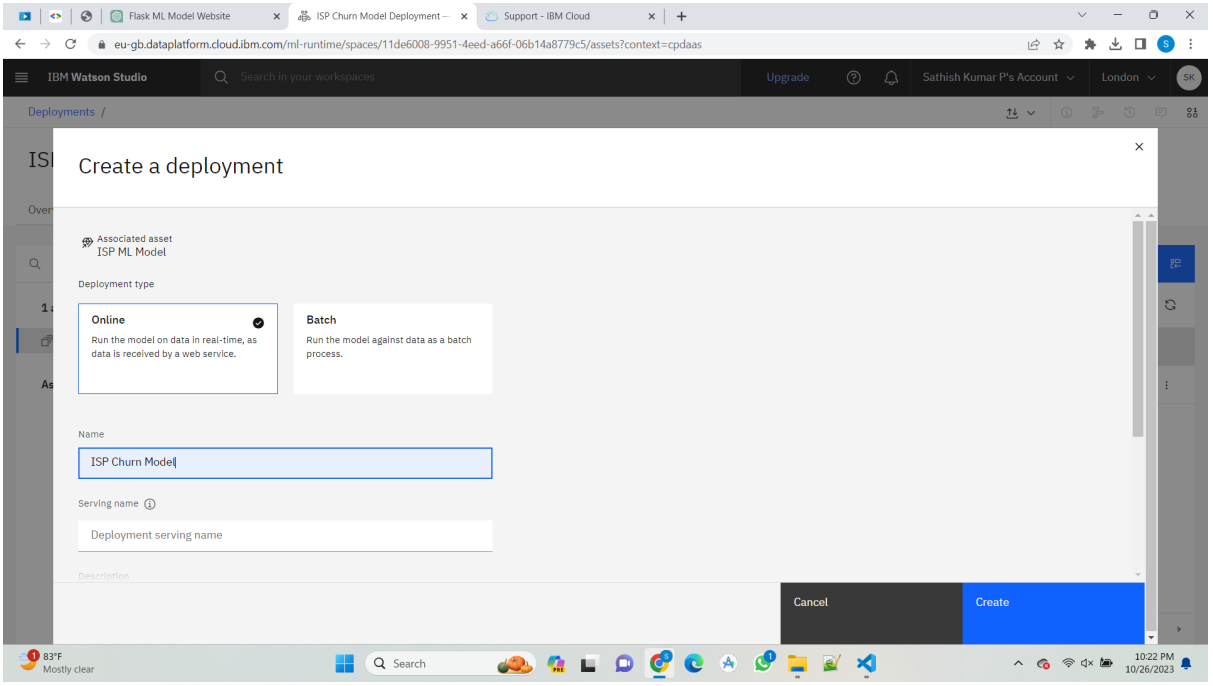
Step:02

Then we need to promote the model to the deployment space



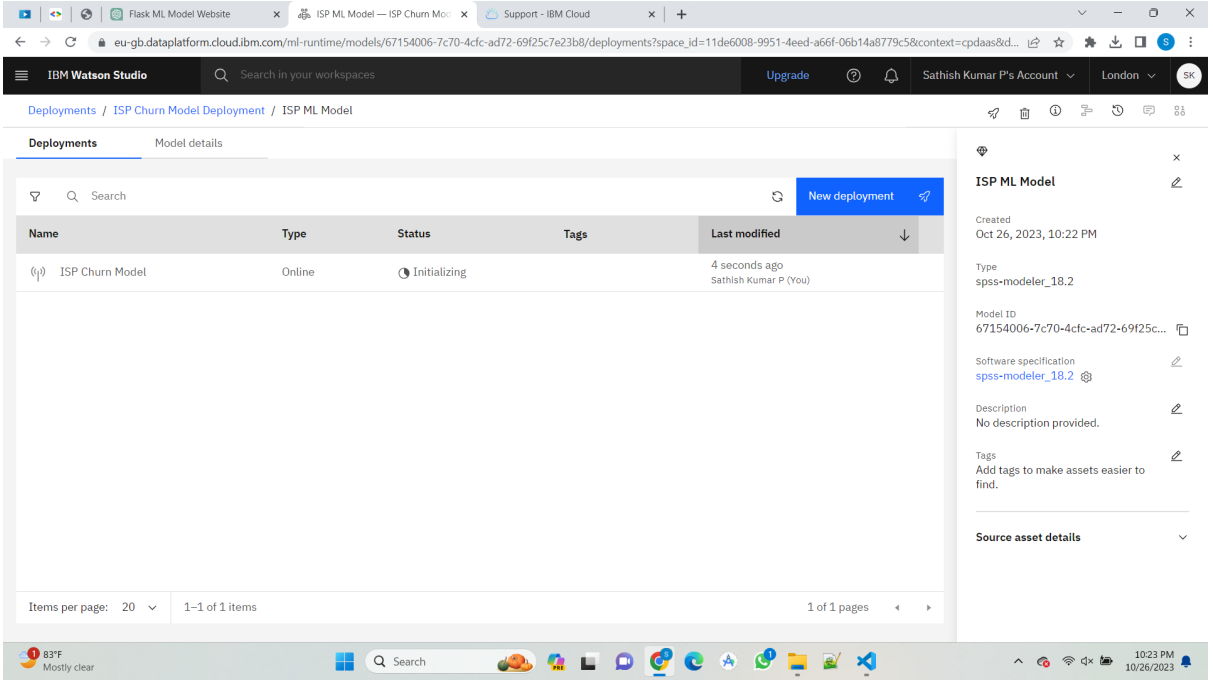
Step:03

The next step is go to the deployment space and click the three dots in the model and click the option deploy as online



Step:04

The it takes few minutes to model be online



Model Integration:

Step:01

Create a web page using html,css and javascript to get the input and predict.

Index.html

```
<!DOCTYPE html>
<html>
<head>
    <title>ISP Churn Model</title>
    <link rel="stylesheet" type="text/css" href="styles.css">
</head>
<body>
    <h1>ISP Churn Model</h1>
    <form method="post" action="/predict">
        <label for="subscription_age">Subscription Age:</label>
        <input type="number" name="subscription_age" required><br>

        <label for="bill_avg">Average Bill Amount:</label>
        <input type="number" name="bill_avg" required><br>

        <label for="service_failure_count">Service Failure Count:</label>
        <input type="number" name="service_failure_count" required><br>

        <label for="download_avg">Average Download Speed:</label>
        <input type="number" name="download_avg" required><br>
```

```
        <label for="upload_avg">Average Upload Speed:</label>
        <input type="number" name="upload_avg" required><br>

        <label for="download_over_limit">Downloads Over Limit (0 for No, 1 for Yes):</label>
        <input type="number" name="download_over_limit" required><br>

        <input type="submit" value="Predict">
    </form>
</body>
</html>
```

Step:02

Then the next step is to create a flask program to integrate the ML model we copy the code from the below deploy model and connect to the web page.

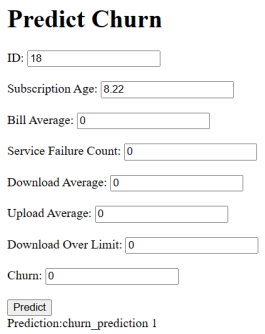
Flask code:

App.py

```
import time
from flask import Flask, render_template, request, jsonify
import requests

app = Flask(__name__)

API_KEY = "iost7rxsblFLvgbMMQkjQvP_xowl0o8J_vochvkLafFN"

ENDPOINT_URL="https://private.eu-de.ml.cloud.ibm.com/ml/v4/deployments/internet_churn
/predictions?version=2021-05-01"

def make_prediction(id, subscription_age, bill_avg, service_failure_count, download_avg,
upload_avg, download_over_limit, churn):
    max_retries = 5
    for attempt in range(max_retries):
        try:
            input_data = {
                "input_data": [
                    {
                        "fields": ["ID", "Subscription_Age", "Bill_Average", "Service_Failure_Count",
"Download_Average", "Upload_Average", "Download_Over_Limit", "Churn"],
                        "values": [[id, subscription_age, bill_avg, service_failure_count,
download_avg, upload_avg, download_over_limit, churn]]
                    }
                ]
            }
```

```
        token_response = requests.post('https://iam.cloud.ibm.com/identity/token',
data={"apikey": API_KEY, "grant_type": 'urn:ibm:params:oauth:grant-type:apikey'})
        token_response.raise_for_status()
        mltoken = token_response.json()["access_token"]


        headers = {'Content-Type': 'application/json', 'Authorization': 'Bearer ' + mltoken}
        response_scoring = requests.post(ENDPOINT_URL, json=input_data,
headers=headers)
        response_scoring.raise_for_status()


        prediction_result = response_scoring.json()

        return prediction_result
    except Exception as e:
        if attempt < max_retries - 1:
            time.sleep(1)
        else:
            return {"error": str(e)}

@app.route('/')
def index():
    return render_template('index.html')

@app.route('/predict', methods=['POST'])
def predict():
    try:
        id = request.form['id']
        subscription_age = float(request.form['subscription_age'])
        bill_avg = float(request.form['bill_avg'])
        service_failure_count = float(request.form['service_failure_count'])
        download_avg = float(request.form['download_avg'])
        upload_avg = float(request.form['upload_avg'])
        download_over_limit = float(request.form['download_over_limit'])
        churn = float(request.form['churn'])

            prediction = make_prediction(id, subscription_age, bill_avg, service_failure_count,
download_avg, upload_avg, download_over_limit, churn)

        return jsonify({"prediction": prediction})

    except Exception as e:
        return jsonify({"error": str(e)})

if __name__ == '__main__':
    app.run(debug=True)
```

Sample output:



Conclusion:

Thus,our ISP churn model is deployed and integrated to the web services successfully.