# **Machine learning model deployment using IBM Watson studio**

**(Phase 5)**

# **Project Summary**

This project is designed to empower individuals and teams with the knowledge and tools to deploy machine learning models effectively using IBM Cloud Watson Studio. We guide you through a structured process, from defining a predictive use case to deploying a model as a web service and integrating it into applications.

## **Purpose**

The primary purpose of this project is to demystify the machine learning model deployment process and provide a hands-on, practical approach to make informed, datadriven decisions in real-time. Whether you are a data scientist, developer, or Al enthusiast, this project aims to equip you with the skills to harness the power of machine learning and predictive analytics.

## **Installation**

1. **Python**: Python is the primary programming language used in most machine learning projects. It provides access to a wide range of libraries and tools for data manipulation, model training, and deployment.
2. **Jupyter Notebooks:** Jupyter notebooks are commonly used for interactive development, data exploration, and documenting your machine learning project.
3. **Scikit-learn:** Scikit-learn is a powerful Python library for machine learning. It provides tools for data preprocessing, model selection, and evaluation.
4. **Pandas**: Pandas is a Python library for data manipulation and analysis. It's often used for data cleaning, transformation, and exploration.
5. **NumPy:** NumPy is a fundamental package for scientific computing in Python. It provides support for working with arrays and matrices, which are essential for machine learning.
6. **IBM Cloud Watson Studio:** This cloudbased platform offers various tools for data science and machine learning, including model training, deployment, and management.
7. **IBM Watson Machine Learning:** Part of the IBM Cloud ecosystem, this service provides capabilities for deploying and serving machine learning models as web services.
8. **Flask**: Flask is a Python web framework commonly used for creating web services and APIs. You might use it to create a RESTful API for your deployed model.
9. **Docker**: Docker is used for containerization, allowing you to package your model and its dependencies for consistent deployment.
10. **Git and GitHub:** Git is a version control system, and GitHub is a platform for hosting and sharing code. You can use these for collaboration and version control.
11. **Cloud Services:** Depending on your project's requirements, you might leverage various IBM Cloud services, such as cloud databases, cloud object storage, and others.
12. **Automated Machine Learning (AutoML)Tools:** IBM Watson AutoAl or other AutoML tools can automate parts of the model selection and tuning process.

**### Required Packages and Libraries**

You can install the necessary packages and libraries for this project using `pip`. Here are the commands to install them:

```bash

# Install Python

# (Make sure you have Python 3.x installed)

# <https://www.python.org/downloads/>

# Install Jupyter Notebooks

Pip install notebook

# Scikit-learn – A powerful machine learning library

Pip install scikit-learn

# Pandas – Data manipulation and analysis

Pip install pandas

# NumPy – Fundamental package for scientific computing

Pip install numpy

# Flask – Python web framework for creating APIs

Pip install flask

# Docker – Containerization tool

# <https://docs.docker.com/get-docker/>

# (Docker installation instructions vary by platform)

# Git – Version control system

# <https://git-scm.com/downloads>

# (Install Git for version control)

### Key Goals

* Learn Predictive Analytics: Understand the fundamentals of predictive analytics and how it can be applied to real-world scenarios.
* Hands-on Experience: Gain practical experience in selecting datasets, training machine learning models, and deploying them as web services.
* Incorporate Predictions: Learn how to seamlessly integrate machine learning models into applications to make realtime predictions.
* Share Knowledge: Create a valuable resource for the community, providing comprehensive documentation and example API requests to guide others in similar endeavors.

Through this project, you will unlock the magic of data-driven insights and develop the skills to make informed decisions based on the outcomes predicted by your machine learning models.

### Objective

The objective of the "Machine Learning Model Deployment with IBM Cloud Watson Studio" project is to empower individuals and teams with the knowledge and tools needed to effectively deploy machine learning models in real-world applications. The project aims to achieve the following goals:

1 . Skill Development: Provide a hands-on learning experience for individuals interested in machine learning model deployment. Participants will gain practical experience in deploying models as web services.

1. Predictive Analytics Proficiency: Enhance predictive analytics skills by exploring a real-world use case and dataset, selecting the right machine learning algorithm, and deploying the model using IBM Cloud Watson Studio.
2. Real-Time Decision-Making: Enable users to make informed decisions in realtime based on predictions generated by the deployed machine learning model.
3. Design Thinking and Innovation: Encourage innovative thinking in addressing predictive use cases and
4. Design Thinking and Innovation: Encourage innovative thinking in addressing predictive use cases and optimizing machine learning models.
5. Contribution and Collaboration: Create an open-source project that welcomes contributions from the community and fosters collaboration among data scientists, developers, and Al enthusiasts.

The project serves as a comprehensive resource for those looking to bridge the gap between model development and real-world deployment, making data-driven insights and predictive analytics accessible to a wider audience.

***Code:***

*# Import necessary libraries and modules*

*Import numpy as np*

*From sklearn.model\_selection import train\_test\_split*

*From sklearn.preprocessing import StandardScaler*

*From sklearn.ensemble import RandomForestClassifier*

*From flask import Flask, request, jsonify*

*# Phase 1: Problem Definition and Design Thinking*

*# Define your predictive use case and dataset (e.g., using randomly generated data)*

*Use\_case = “Predict Customer Churn”*

*# Generate random data for demonstration*

*Np.random.seed(0)*

*N\_samples = 1000*

*N\_features = 5*

*X = np.random.rand(n\_samples, n\_features)*

*Y = np.random.randint(2, size=n\_samples)*

*# Phase 3: Development 1*

*# Preprocess the data (e.g., scale features)*

*# Split the data into training and testing sets*

*# Train a machine learning model (e.g., Random Forest)*

*X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)*

*Scaler = StandardScaler()*

*X\_train = scaler.fit\_transform(X\_train)*

*X\_test = scaler.transform(X\_test)*

*Model = RandomForestClassifier(n\_estimators=100, random\_state=42)*

*Model.fit(X\_train, y\_train)*

*# Phase 4: Development 2*

*# Create a Flask web application*

*App = Flask(\_\_name\_\_)*

*# Define an endpoint for making predictions*

*@app.route(‘/predict’, methods=[‘POST’])*

*Def predict():*

*# Get input data from the request*

*Data = request.get\_json()*

*Features = data[‘features’]*

*# Perform any necessary preprocessing on the input data*

*Features = scaler.transform([features])*

*# Make predictions using the trained model*

*Prediction = model.predict(features)*

*# Return the prediction as JSON response*

*Return jsonify({‘prediction’: prediction.tolist()})*

*If \_\_name\_\_ == ‘\_\_main\_\_’:*

*App.run(debug=True)*