Phase 5 Documentation & Submission

Date	**28-10-2023**
Team ID	**4146**
Project Name	**Machine learning model deployment with IBM Watson studio**

Project Title: Heart Disease Prediction Model Deployment in IBM Watson Studio

Table of Contents

1.Introduction

In an era driven by data and technology, healthcare is undergoing a transformative journey. This project represents our endeavor to harness the power of machine learning for the early detection of heart disease. We trained and evaluated three distinct machine learning models - Gradient Boosting, Support Vector Machine (SVM), and Random Forest - with the objective of developing an accurate prediction tool for heart disease.

This document provides an overview of our comprehensive project, which encompassed the development and deployment of a heart disease prediction model. The project was executed in three key phases: Model Development, IBM Cloud Watson Studio Deployment, Flask Integration and API Endpoint Creation.

2.Problem Statement

Objective: Deploy a machine learning model with IBM Watson Studio as a web service that can predict heart disease.

Data: We have a dataset containing various features for predicting the heart disease (e.g., age, blood pressure, cholesterol levels, etc). This data will be used to train and evaluate our machine learning model.

3.Project Overview

Model Development:

- We initially developed the heart disease prediction model using Gradient Boosting. The model was trained and evaluated for its predictive accuracy using a heart disease dataset.

IBM Cloud Watson Studio Deployment:

- We leveraged the capabilities of IBM Cloud Watson Studio to deploy the trained model. This cloud-based environment provides robust tools for model deployment and management.

Flask Integration:

- To make the model accessible over the internet, we integrated it into a Flask application. Flask is a lightweight web framework that allows us to create web-based interfaces and provide API endpoints for model predictions.

API Endpoint Creation:

- We defined API endpoints in our Flask application to enable communication with the deployed model. These endpoints accept input data, make predictions using the model, and return results to the users.

4.Objective

Our goal is to design and deploy a scalable and accessible web service that leverages the power of machine learning to predict the risk of heart disease in individuals based on their medical and lifestyle data. The deployment will use IBM Watson Studio, an integrated environment for data science and machine learning, to provide a user-friendly interface for patients to assess heart disease risk quickly and accurately.

5.Literature Survey:

1.“Clinical Implication of Machine Learning Based Cardiovascular Disease Prediction Using IBM Auto AI Service ”, M. Nirmala [2022]

In this project, a Random Forest Classifier model is used with Auto AI to create a web application via Node Red. The application displays predictions for heart failure, a complex disease influenced by various risk factors. The model uses 10 attributes for prediction and leverages IBM AUTO AI. Cardiovascular disease is a major cause of death, and data analytics techniques are employed to predict its occurrence. The model achieves 87% accuracy with Random Forest and 79% with Logistic Regression when integrated into the Node Red application. Further improvements are needed for generalization and user-friendliness.

2. “ML Enabled WhatsApp Chatbot using IBM Watson”, Manasi Chhibber[2022]

The work aimed to create a chatbot for loan applications using machine learning to improve speed, accuracy, and reduce biases. It automated the process with IBM Watson services, including Watson Studio, Cloud Functions, and Watson Assistant for WhatsApp integration. The machine learning phase used AutoAI, and XGBoost was the chosen model. Deployment involved creating a space and integrating with Watson Assistant. WhatsApp integration was done via Twilio. The chatbot streamlines loan applications and offers risk assessments, improving accessibility via WhatsApp.

3. “An approach for predicting heart failure rate using IBM Auto AI Service”, Krishna Priya G, Suganthi S T, M, Vijipriya G, Nirmala M [2021]

This paper explores IBM's AutoAI-based machine learning model for heart failure prediction, emphasizing its automation and accuracy. The system deploys a classification model using gradient boosting, visualized through NodeRED. It streamlines heart failure prediction without manual data preprocessing or feature extraction. The service is part of IBM Watson Studio, enabling collaborative data science and AI work without coding. The best model performance achieved an accuracy of 0.874, suggesting a shift from detection to prevention. Moreover, automatic and fast applications like chatbots, natural language disambiguation, and sensor-based cloud apps can be developed.

4. “AI Algorithm System for Prediction of Diabetes Using Progressive Web App and IBM Cloud”, Dr. Mohammed Abdul Raheem , Shaik Ehetesham , Mohammad Faiz Ahmed Subhani , Sayed Abdul Zakir [2020]

The research aims to design a model for early diabetes prediction and overall subject well-being. In this study, researchers developed Machine Learning models using historical data and created a Progressive Web app to detect diabetes in its early stages. The process involved problem identification, fieldwork, model building, integration with IBM Cloud using Flask and Docker, testing, and the creation of the Progressive Web App. Future improvements could include adding parameters like hereditary and gestational diabetes to enhance prediction accuracy and incorporating automatic location detection for convenient patient referrals to nearby diagnostic centers.

5. “Early Health Prediction System for ICU Patient using Machine Learning and Cloud Computing”, Asif Ahmed Neloy, Muhammad Shafayat Oshman, Md. Monzurul Islam, Md Julhas Hossain and Zunayeed Bin Zahir [2019]

This project aims to create a real-time feedback system for ICU patient care in hospitals using machine learning and cloud computing. The paper proposes a generic architecture and classification model for monitoring ICU patient health. The problem addressed is the lack of technological support in the healthcare sector in Bangladesh, leading to inefficient patient care during emergencies. The solution allows remote monitoring of patient vitals by doctors, leveraging machine learning and cloud computing. The project achieved over 90% success with IBM Cloud and plans to expand to a larger scale, including an embedded system for real-time data collection from ICU machines, improving overall accuracy.

6.Design Thinking:

6.1. Data Collection and Preparation

- Collect a diverse dataset related to heart disease, ensuring it encompasses a wide range of relevant features.
- Innovatively clean and preprocess the data, considering advanced data imputation techniques and feature engineering to enhance model accuracy.

6.2. Feature Engineering

- Use domain knowledge to select relevant features for heart disease prediction.
- Create new features or transformations that capture complex relationships within the data.
- Implement feature selection techniques to identify the most important predictors.

6.3. Model Development

- Utilize advanced machine learning algorithms such as ensemble models (e.g., Random Forest, Gradient Boosting) and deep learning models (e.g., neural networks) for heart disease prediction.
- Implement a hyperparameter tuning strategy, employing techniques like Bayesian Optimization or Genetic Algorithms to optimize the model's performance.

6.4. Model Evaluation and Deployment

- Develop an innovative approach for model evaluation, considering not only accuracy but also interpretability and explainability to ensure trust and understanding of the predictions.
- Deploy the trained Model in IBM Watson Studio.

6.5. Web Service Development using IBM Watson Studio

- Use IBM Watson Studio to deploy the heart disease prediction model as a web service in a secure and scalable manner.
- Leverage Watson Machine Learning for model deployment, allowing users to access the prediction service seamlessly.

6.6. User Interface and Experience Enhancement

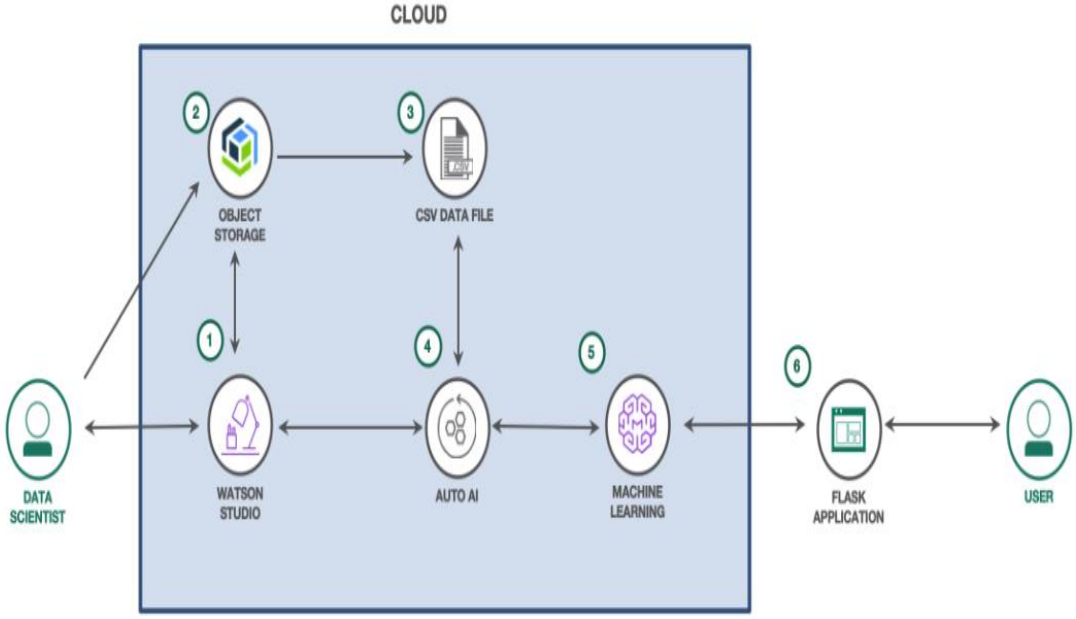
- Design an intuitive, user-friendly web interface that allows users to input their health data and receive predictions for heart disease risk.

- Incorporate innovative features like real-time feedback during data input, visually appealing graphs to display prediction probabilities, and personalized health recommendations based on the prediction outcome.

6.8. Testing and Continuous Improvement

- Conduct rigorous testing to validate the functionality, performance, and user experience of the web service.

- Gather user feedback and analytics to continuously improve the model and the user interface, implementing updates based on user needs and preferences.



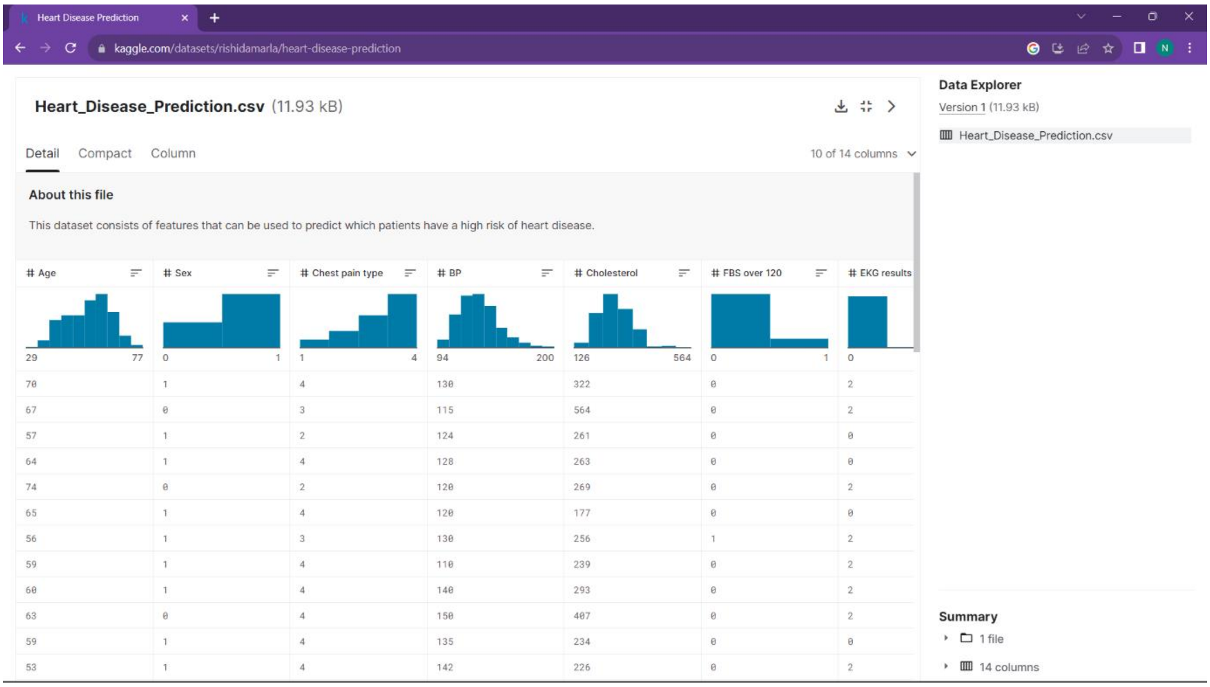
7.Development Phases:

The project progressed through the following key phases:

7.1.Data Collection and Preprocessing

- Acquired a dataset containing health-related features and heart disease labels from kaggle

- Preprocessed the data by handling missing values, encoding categorical variables, and scaling numeric features.



Heart_Disease_Prediction.csv (11.93 kB)

About this file

This dataset consists of features that can be used to predict which patients have a high risk of heart disease.

# Age	# Sex	# Chest pain type	# BP	# Cholesterol	# FBS over 120	# EKG results
70	1	4	130	322	0	2
67	0	3	115	564	0	2
57	1	2	124	261	0	0
64	1	4	128	263	0	0
74	0	2	120	269	0	2
65	1	4	120	177	0	0
56	1	3	130	256	1	2
59	1	4	110	239	0	2
60	1	4	140	293	0	2
63	0	4	150	407	0	2
59	1	4	135	234	0	0
53	1	4	142	226	0	2

Data Explorer

Version 1 (11.93 kB)

Heart_Disease_Prediction.csv

Summary

1 file

14 columns

7.2.Model Development:

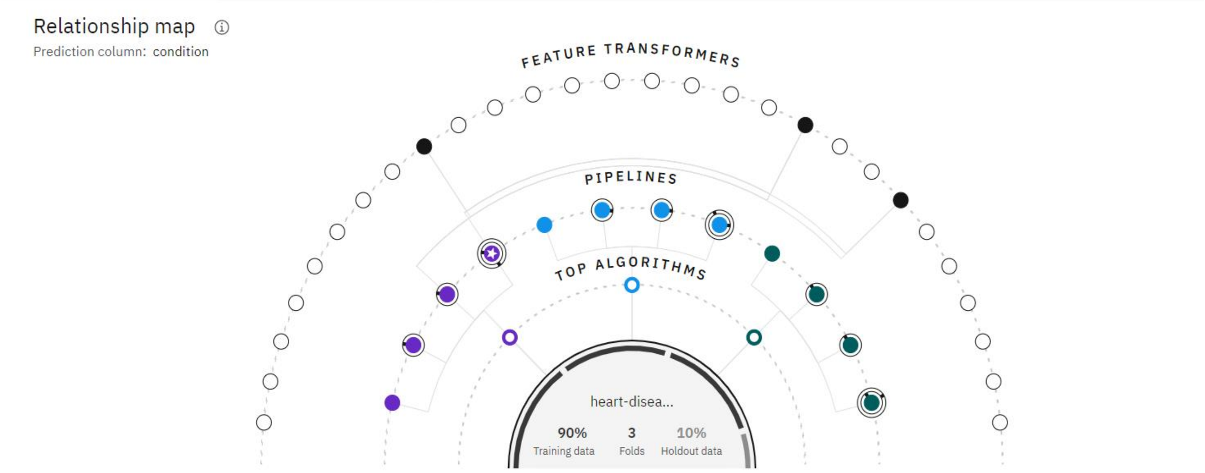
7.2.1.ML Algorithm Selection

For our heart disease prediction model, we selected three powerful classifiers:

- **Gradient Boosting:** Gradient Boosting is an ensemble learning method that combines the predictions of multiple decision trees, iteratively improving model performance.
- **Snap Support Vector Machine (SVM):** SVM is a robust and versatile classifier that works well for both linear and non-linear classification problems.
- **Random Forest**: Random Forest is another ensemble method that combines multiple decision trees to make predictions. It is known for its ability to handle complex datasets and maintain good accuracy.

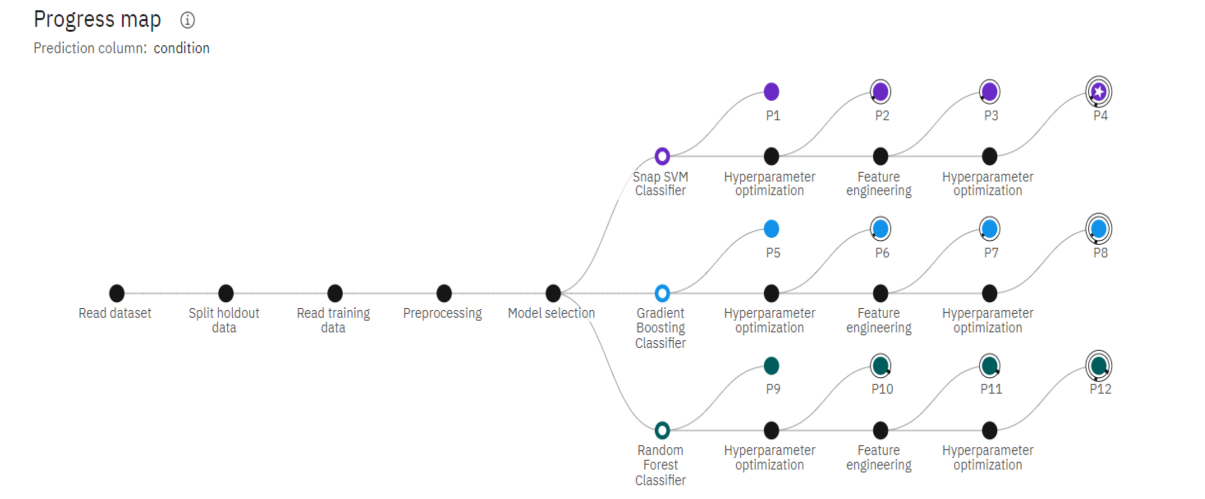
7.2.2.Model Training

- We trained each classifier using the preprocessed dataset and conducted cross-validation to optimize hyperparameters.
- The models were evaluated based on performance metrics such as accuracy, precision, recall, and F1-score.



Relationship map

Prediction column: condition



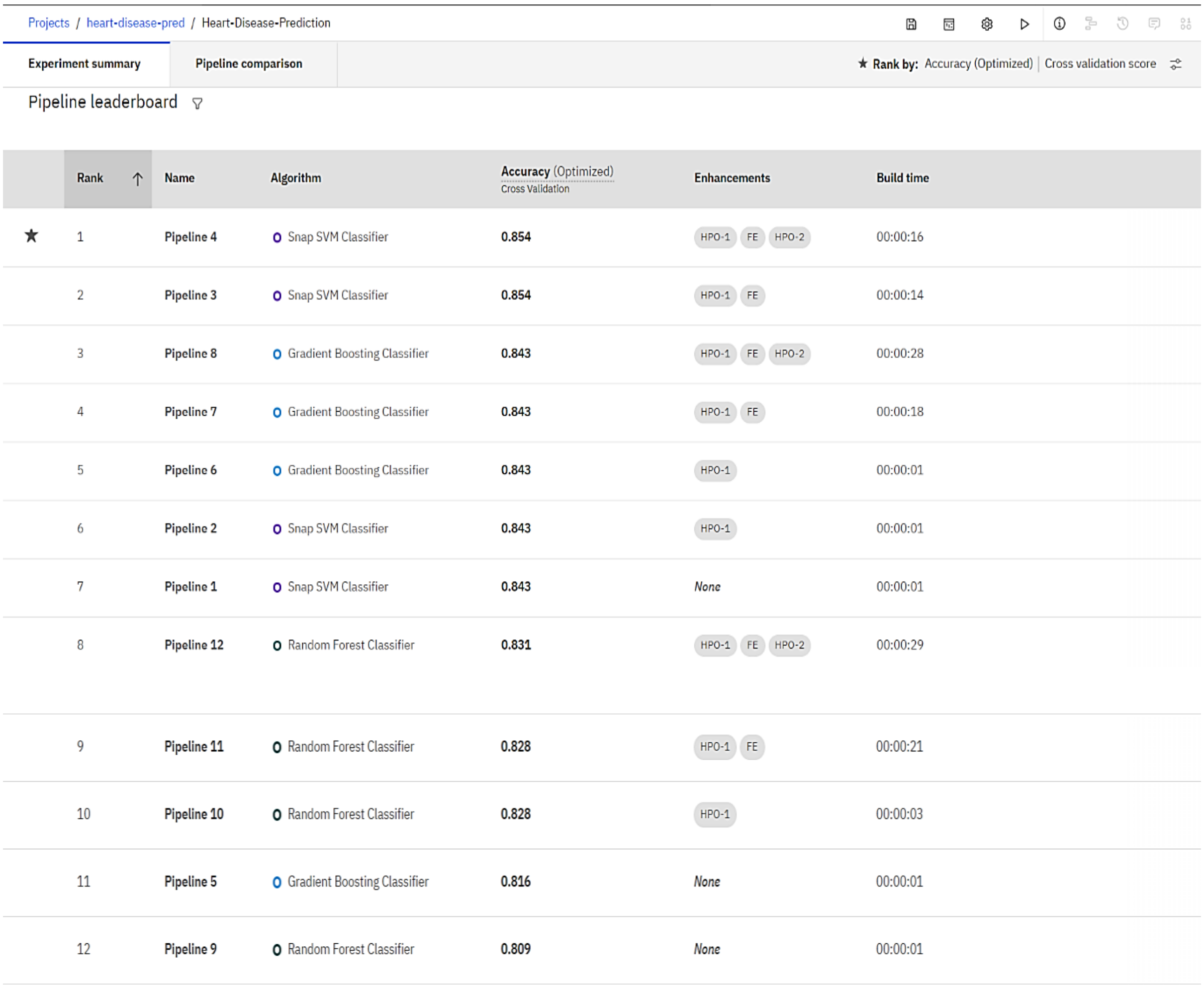
Progress map

Prediction column: condition

7.2.3.Model Evaluation

- We used various evaluation metrics such as accuracy, precision, recall, F1-score, and ROC AUC to assess the performance of each model.

- The models were tested rigorously using cross-validation and an independent test dataset.



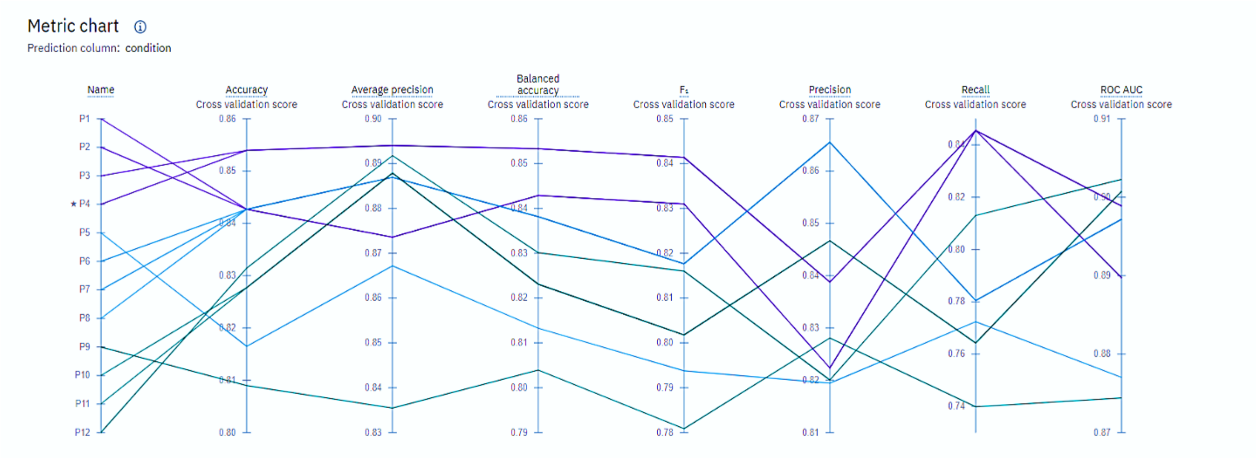
Projects / heart-disease-pred / Heart-Disease-Prediction

Experiment summary | Pipeline comparison

★ Rank by: Accuracy (Optimized) | Cross validation score

Pipeline leaderboard

	Rank	Name	Algorithm	Accuracy (Optimized) Cross Validation	Enhancements	Build time
★	1	Pipeline 4	Snap SVM Classifier	0.854	HPO-1 FE HPO-2	00:00:16
	2	Pipeline 3	Snap SVM Classifier	0.854	HPO-1 FE	00:00:14
	3	Pipeline 8	Gradient Boosting Classifier	0.843	HPO-1 FE HPO-2	00:00:28
	4	Pipeline 7	Gradient Boosting Classifier	0.843	HPO-1 FE	00:00:18
	5	Pipeline 6	Gradient Boosting Classifier	0.843	HPO-1	00:00:01
	6	Pipeline 2	Snap SVM Classifier	0.843	HPO-1	00:00:01
	7	Pipeline 1	Snap SVM Classifier	0.843	None	00:00:01
	8	Pipeline 12	Random Forest Classifier	0.831	HPO-1 FE HPO-2	00:00:29
	9	Pipeline 11	Random Forest Classifier	0.828	HPO-1 FE	00:00:21
	10	Pipeline 10	Random Forest Classifier	0.828	HPO-1	00:00:03
	11	Pipeline 5	Gradient Boosting Classifier	0.816	None	00:00:01
	12	Pipeline 9	Random Forest Classifier	0.809	None	00:00:01



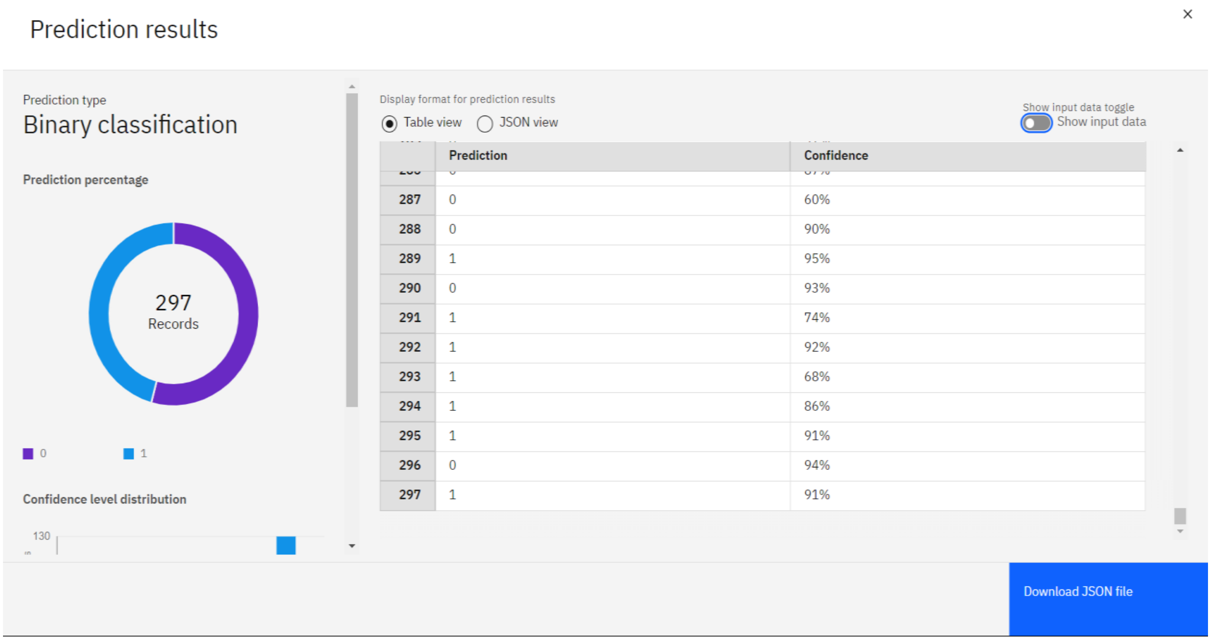
Metric chart

Prediction column: condition

7.2.4.Results

In this phase, we focused on creating and evaluating heart disease prediction models using three diverse machine learning algorithms:

- **Gradient Boosting**: A powerful ensemble learning technique was employed to build a robust predictive model. Through rigorous training and testing, this classifier emerged as the top performer in terms of predictive accuracy.
- **Support Vector Machine (SVM)**: SVM, known for its ability to handle complex decision boundaries, was another model considered. However, it was outperformed by Gradient Boosting during testing.
- **Random Forest**: The Random Forest classifier, with its bagging approach, was the third model in our lineup. While it showed promise, it did not surpass Gradient Boosting's performance.



Prediction results

Prediction type
Binary classification

Prediction percentage

297 Records

🟪 0 🟦 1

Confidence level distribution

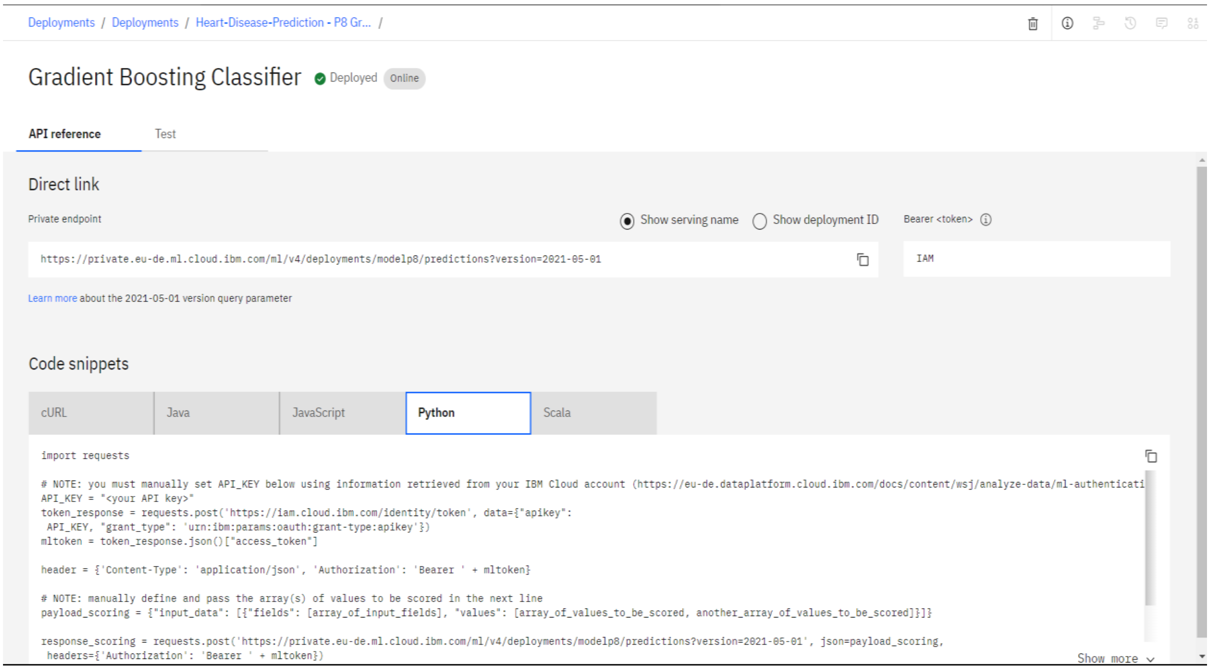
Display format for prediction results
● Table view ○ JSON view

Show input data toggle
Show input data

	Prediction	Confidence
287	0	60%
288	0	90%
289	1	95%
290	0	93%
291	1	74%
292	1	92%
293	1	68%
294	1	86%
295	1	91%
296	0	94%
297	1	91%

7.3.IBM Cloud Watson Studio Deployment

- Exported the Gradient Boosting model and deployed it on IBM Cloud Watson Studio.
- Configured deployment settings, including scalability and security options.



https://private.eu-de.ml.cloud.ibm.com/ml/v4/deployments/modelp8/predictions?version=2021-05-01

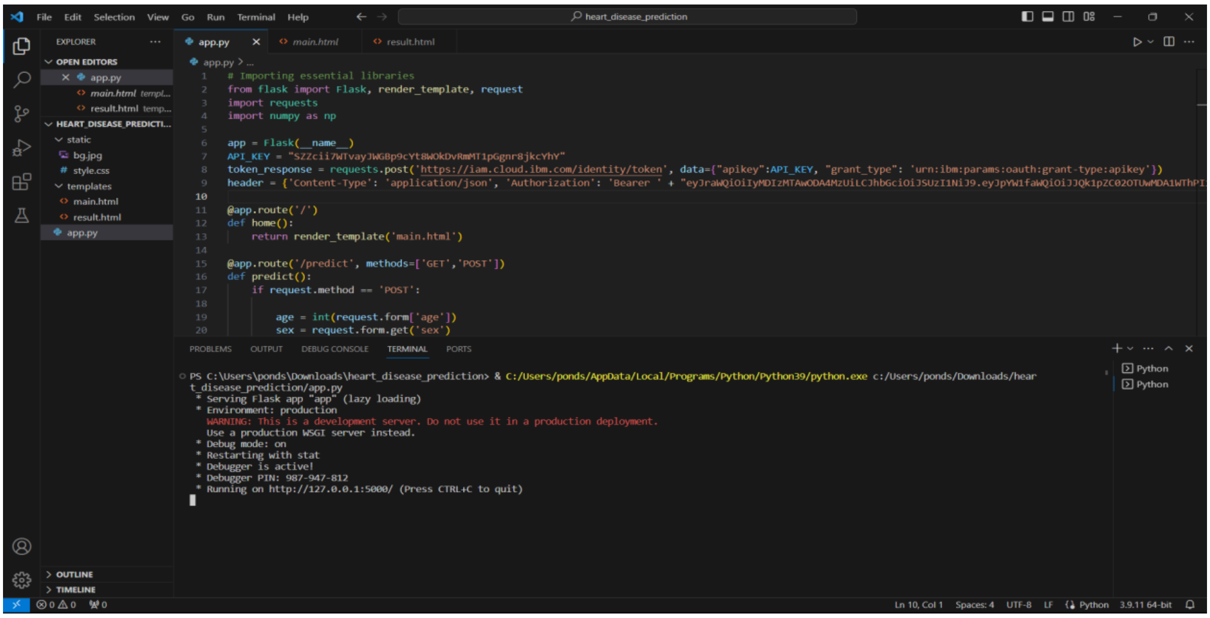
IAM

Learn more about the 2021-05-01 version query parameter

```
import requests

# NOTE: you must manually set API_KEY below using information retrieved from your IBM Cloud account (https://eu-de.dataplatform.cloud.ibm.com/docs/content/wsj/analyze-data/ml-authenticati
API_KEY = "<your API key>"
token_response = requests.post('https://iam.cloud.ibm.com/identity/token', data={"apikey":
 API_KEY, "grant_type": 'urn:ibm:params:oauth:grant-type:apikey'})
mltoken = token_response.json()["access_token"]

header = {'Content-Type': 'application/json', 'Authorization': 'Bearer ' + mltoken}

# NOTE: manually define and pass the array(s) of values to be scored in the next line
payload_scoring = {"input_data": [{"fields": [array_of_input_fields], "values": [array_of_values_to_be_scored, another_array_of_values_to_be_scored]}]}

response_scoring = requests.post('https://private.eu-de.ml.cloud.ibm.com/ml/v4/deployments/modelp8/predictions?version=2021-05-01', json=payload_scoring,
 headers={'Authorization': 'Bearer ' + mltoken})
```

7.4.Flask Integration

- Developed a Flask application with API endpoints to interact with the deployed model.
- Implemented data preprocessing and post-processing to ensure accurate predictions.

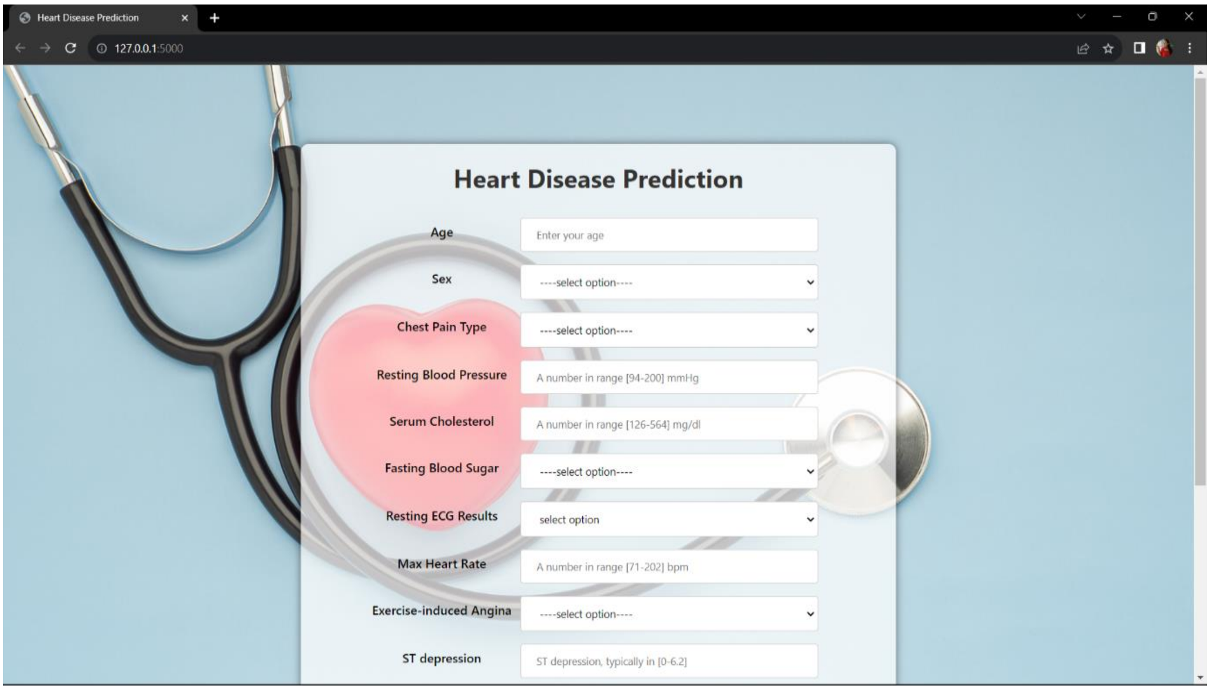


```
# Importing essential libraries
from flask import Flask, render_template, request
import requests
import numpy as np

app = Flask(__name__)
API_KEY = "SZZcii7WTvayJWGBp9cYt8WOkDvRmMT1pGgnr8jkcYhY"
token_response = requests.post('https://iam.cloud.ibm.com/identity/token', data={"apikey":API_KEY, "grant_type": 'urn:ibm:params:oauth:grant-type:apikey'})
header = {'Content-Type': 'application/json', 'Authorization': 'Bearer ' + "eyJraWQiOiIyMDIzMTAwODA4MzUiLCJhbGciOiJSUzI1NiJ9.eyJpYW1faWQiOiJJQk1pZC02OTUwMDA1WThPI

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

@app.route('/predict', methods=['GET','POST'])
def predict():
    if request.method == 'POST':

        age = int(request.form['age'])
        sex = request.form.get('sex')
```

```
PS C:\Users\ponds\Downloads\heart_disease_prediction> & C:/Users/ponds/AppData/Local/Programs/Python/Python39/python.exe c:/Users/ponds/Downloads/hear
t_disease_prediction/app.py
 * Serving Flask app "app" (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: on
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 987-947-812
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

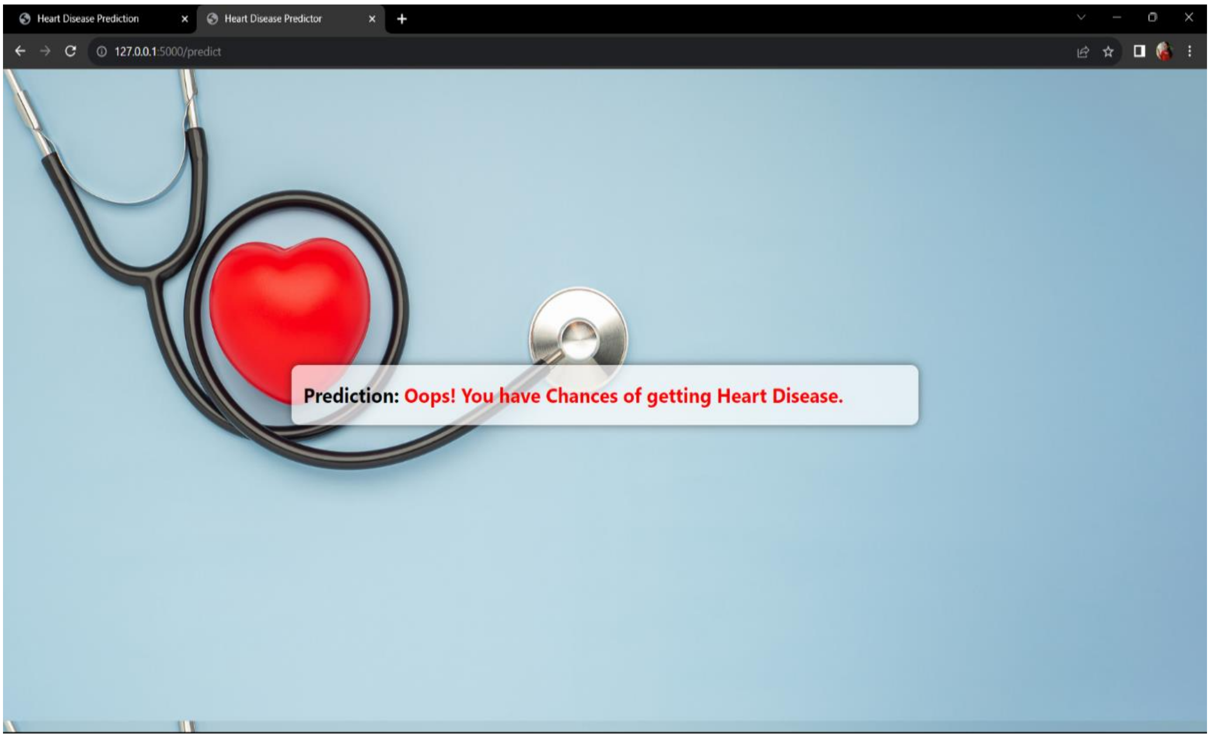
7.5.User Interface Design

- Designed a user-friendly web interface using HTML and CSS.
- Created a form for users to input health-related data.
- Model prediction results displayed to the user in a clear and understandable format.



Heart Disease Prediction

Age	Enter your age
Sex	----select option----
Chest Pain Type	----select option----
Resting Blood Pressure	A number in range [94-200] mmHg
Serum Cholesterol	A number in range [126-564] mg/dl
Fasting Blood Sugar	----select option----
Resting ECG Results	select option
Max Heart Rate	A number in range [71-202] bpm
Exercise-induced Angina	----select option----
ST depression	ST depression, typically in [0-6.2]



Prediction: Oops! You have Chances of getting Heart Disease.

8.Platform Overview:

IBM Cloud Watson Studio

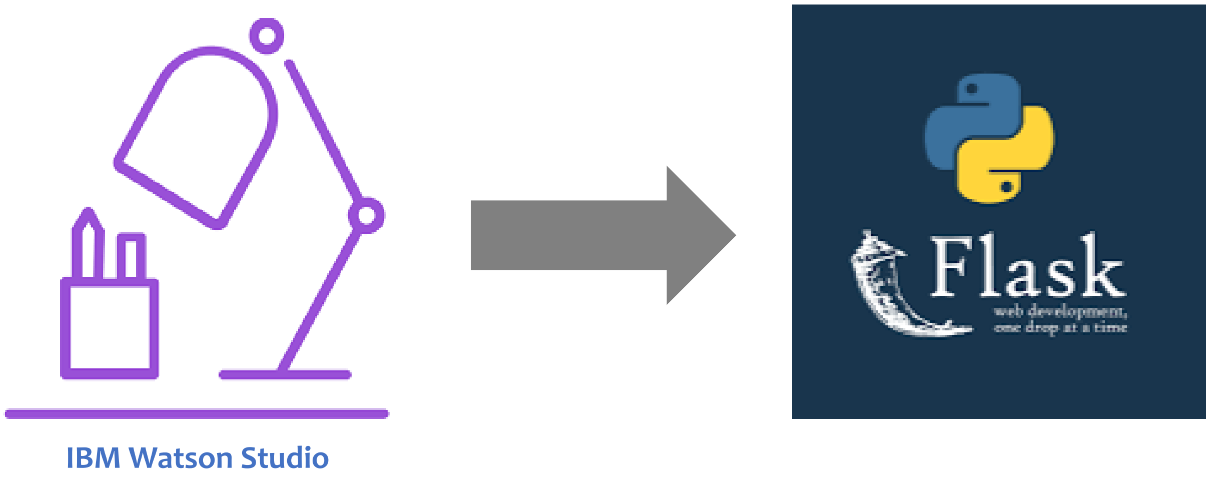
IBM Cloud Watson Studio offered a robust and user-friendly environment for our project. It provided tools for data preprocessing, model development, and deployment. The platform's key features included:

- **Data Preparation**: Watson Studio's data preparation tools allowed us to clean, preprocess, and format the heart disease dataset for training.
- **Model Development**: We trained and evaluated multiple machine learning models in Watson Studio, enabling a streamlined development process.
- **Model Deployment**: Watson Studio offered straightforward deployment options for our chosen Gradient Boosting model.

Flask Application

The Flask application served as the bridge between the deployed model and end-users. The application featured:

- **API Endpoints**: We created custom API endpoints that accepted user input, processed it, and returned predictions from the deployed model.
- **User Interface (UI)**: We designed a user-friendly UI using HTML and CSS, enabling users to interact with the model easily.
- **Data Processing**: The Flask application handled data preprocessing and post-processing, ensuring meaningful and accurate predictions.



9.Technical Implementation Details:

Model Development

- Gradient Boosting, SVM, and Random Forest classifiers were trained using the heart disease dataset.
- Multiple evaluation metrics, including accuracy, precision, recall, F1-score, and ROC AUC, were used to assess the models.

Deployment on IBM Cloud Watson Studio

- The Gradient Boosting model was exported and deployed within the Watson Studio environment.

Flask Integration

- A Flask application was created to host the deployed model.
- Custom API endpoints were established for user interaction.

User Interface (UI)

- HTML and CSS were used to create a visually appealing user interface for data input.

Testing and Refinement

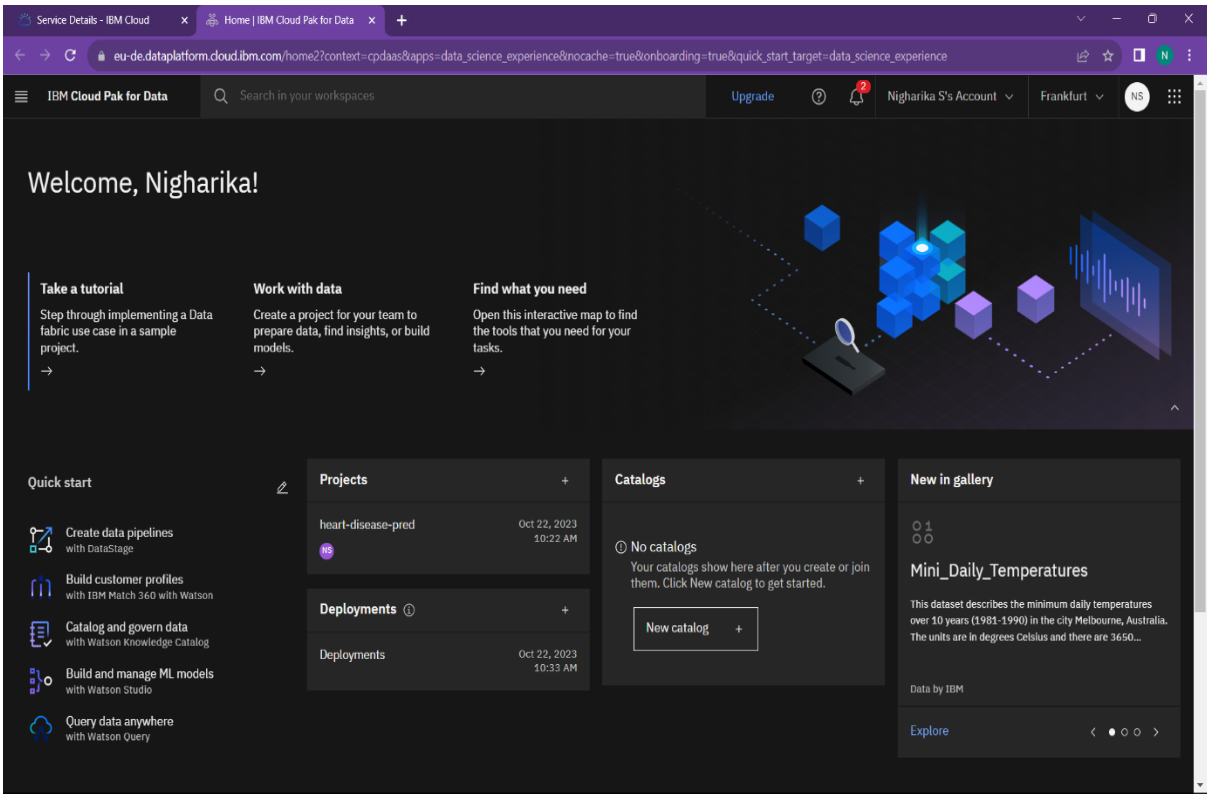
- Rigorous testing was conducted to ensure that the system functioned as intended.

Documentation and User Guidance

- Comprehensive documentation was created to guide users in utilizing the system effectively.

10.Project Development steps and Screenshots:

Step 1: Account creation and create a new project in IBM Watson studio



Service Details - IBM Cloud | Home | IBM Cloud Pak for Data

eu-de.dataplatform.cloud.ibm.com/home2?context=cpdaas&apps=data_science_experience&nocache=true&onboarding=true&quick_start_target=data_science_experience

IBM Cloud Pak for Data | Search in your workspaces | Upgrade | Nigharika S's Account | Frankfurt

Welcome, Nigharika!

Take a tutorial
Step through implementing a Data fabric use case in a sample project.

Work with data
Create a project for your team to prepare data, find insights, or build models.

Find what you need
Open this interactive map to find the tools that you need for your tasks.

Quick start

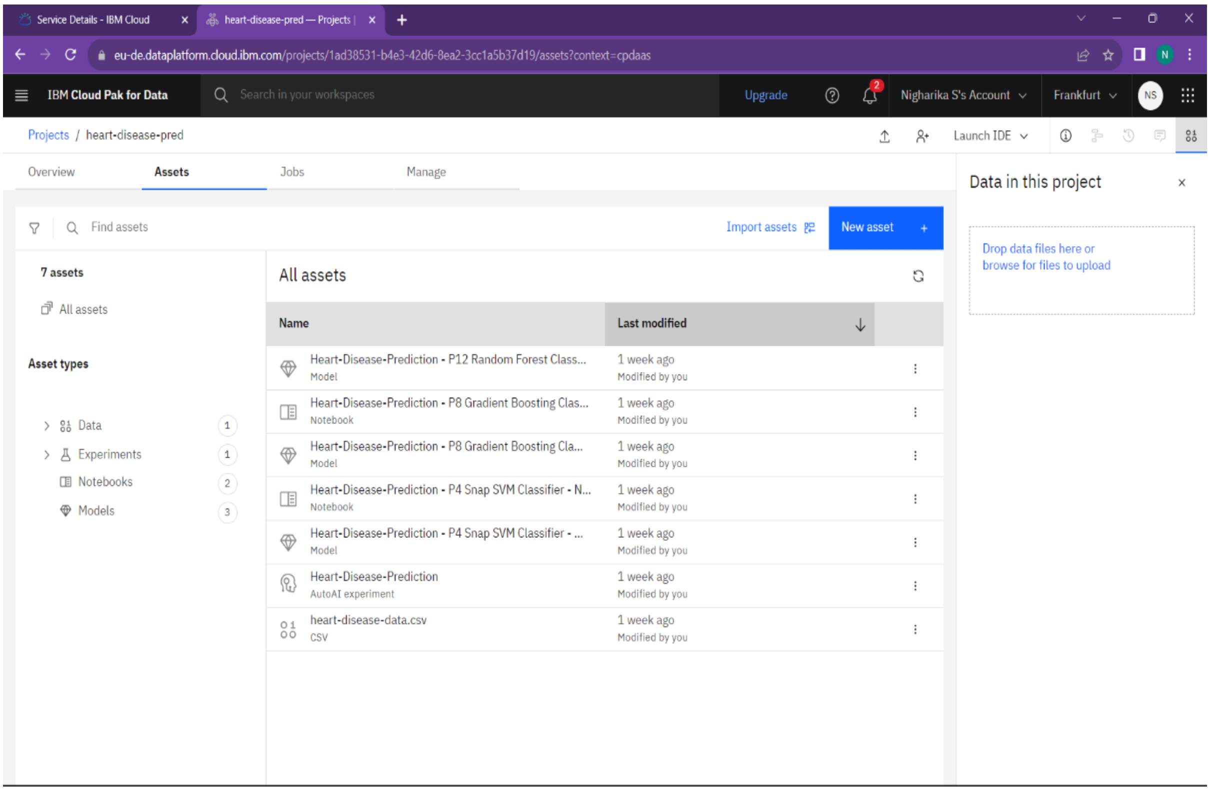
- Create data pipelines with DataStage
- Build customer profiles with IBM Match 360 with Watson
- Catalog and govern data with Watson Knowledge Catalog
- Build and manage ML models with Watson Studio
- Query data anywhere with Watson Query

Projects
heart-disease-pred — Oct 22, 2023 10:22 AM

Deployments
Deployments — Oct 22, 2023 10:33 AM

Catalogs
No catalogs
Your catalogs show here after you create or join them. Click New catalog to get started.
New catalog +

New in gallery
Mini_Daily_Temperatures
This dataset describes the minimum daily temperatures over 10 years (1981-1990) in the city Melbourne, Australia. The units are in degrees Celsius and there are 3650...
Data by IBM
Explore



Service Details - IBM Cloud | heart-disease-pred — Projects |

eu-de.dataplatform.cloud.ibm.com/projects/1ad38531-b4e3-42d6-8ea2-3cc1a5b37d19/assets?context=cpdaas

IBM Cloud Pak for Data | Search in your workspaces | Upgrade | Nigharika S's Account | Frankfurt

Projects / heart-disease-pred | Launch IDE

Overview | Assets | Jobs | Manage

Find assets | Import assets | New asset +

7 assets
All assets

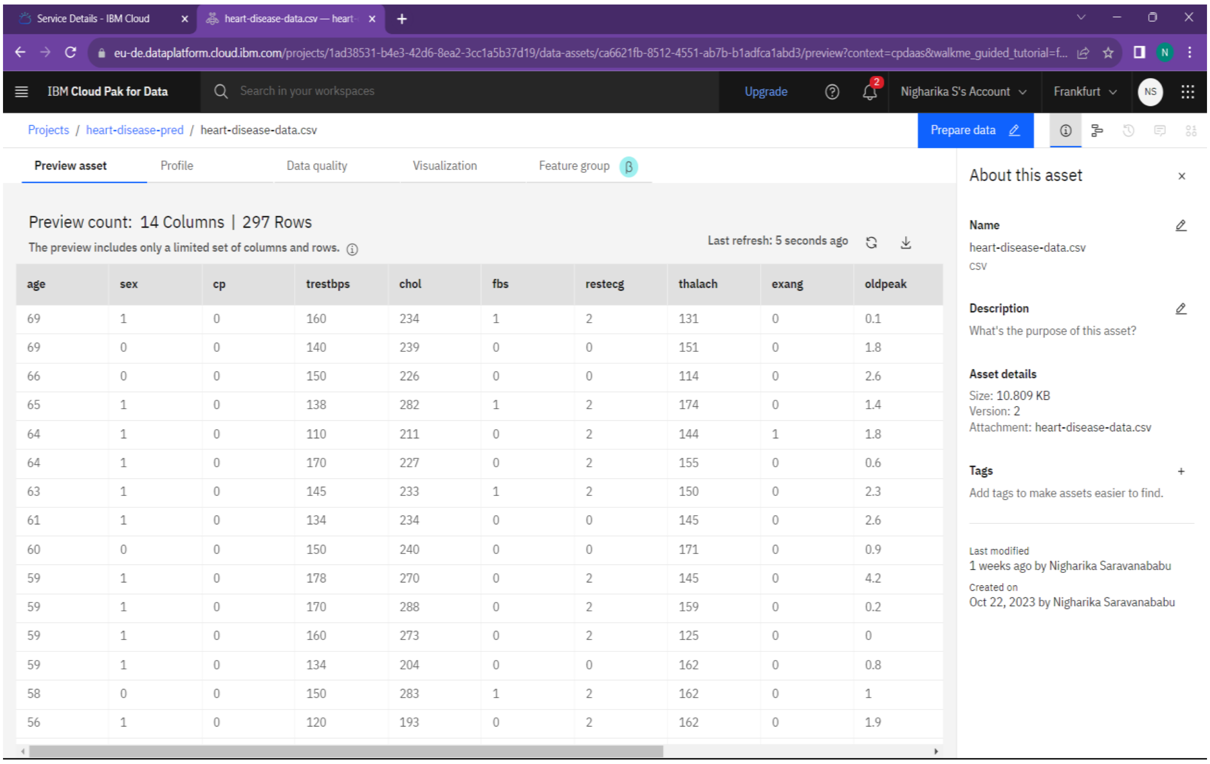
Asset types
- Data 1
- Experiments 1
- Notebooks 2
- Models 3

All assets

Name	Last modified
Heart-Disease-Prediction - P12 Random Forest Class... Model	1 week ago Modified by you
Heart-Disease-Prediction - P8 Gradient Boosting Clas... Notebook	1 week ago Modified by you
Heart-Disease-Prediction - P8 Gradient Boosting Cla... Model	1 week ago Modified by you
Heart-Disease-Prediction - P4 Snap SVM Classifier - N... Notebook	1 week ago Modified by you
Heart-Disease-Prediction - P4 Snap SVM Classifier - ... Model	1 week ago Modified by you
Heart-Disease-Prediction AutoAI experiment	1 week ago Modified by you
heart-disease-data.csv CSV	1 week ago Modified by you

Data in this project
Drop data files here or browse for files to upload

Step 2 : Choose the Dataset for to train the model


Service Details - IBM Cloud
heart-disease-data.csv — heart
eu-de.dataplatform.cloud.ibm.com/projects/1ad38531-b4e3-42d6-8ea2-3cc1a5b37d19/data-assets/ca6621fb-8512-4551-ab7b-b1adfca1abd3/preview?context=cpdaas&walkme_guided_tutorial=f...
IBM Cloud Pak for Data
Search in your workspaces
Upgrade
Nigharika S's Account
Frankfurt
NS
Projects / heart-disease-pred / heart-disease-data.csv
Prepare data
Preview asset
Profile
Data quality
Visualization
Feature group

Preview count: 14 Columns | 297 Rows

The preview includes only a limited set of columns and rows.

Last refresh: 5 seconds ago

age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak
69	1	0	160	234	1	2	131	0	0.1
69	0	0	140	239	0	0	151	0	1.8
66	0	0	150	226	0	0	114	0	2.6
65	1	0	138	282	1	2	174	0	1.4
64	1	0	110	211	0	2	144	1	1.8
64	1	0	170	227	0	2	155	0	0.6
63	1	0	145	233	1	2	150	0	2.3
61	1	0	134	234	0	0	145	0	2.6
60	0	0	150	240	0	0	171	0	0.9
59	1	0	178	270	0	2	145	0	4.2
59	1	0	170	288	0	2	159	0	0.2
59	1	0	160	273	0	2	125	0	0
59	1	0	134	204	0	0	162	0	0.8
58	0	0	150	283	1	2	162	0	1
56	1	0	120	193	0	2	162	0	1.9

About this asset

Name
heart-disease-data.csv
csv

Description
What's the purpose of this asset?

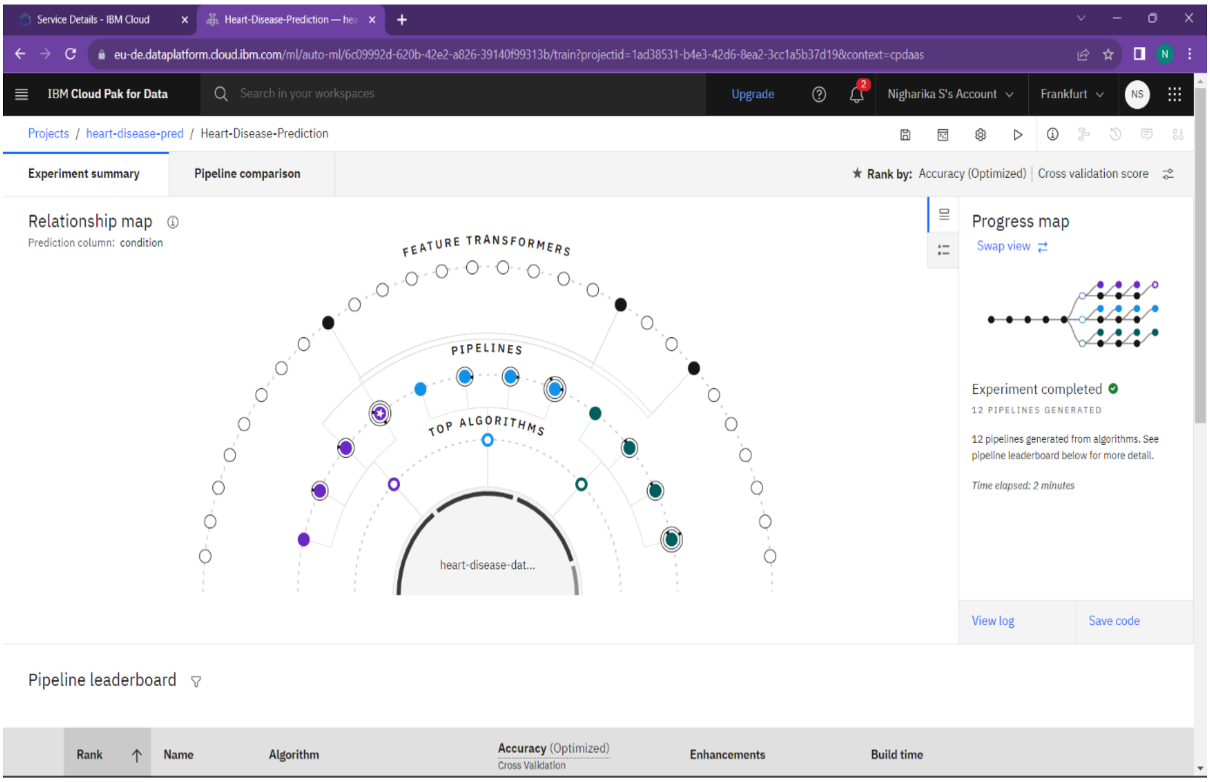
Asset details
Size: 10.809 KB
Version: 2
Attachment: heart-disease-data.csv

Tags
Add tags to make assets easier to find.

Last modified
1 weeks ago by Nigharika Saravanababu

Created on
Oct 22, 2023 by Nigharika Saravanababu

Step 3 : Train the Model


Service Details - IBM Cloud
Heart-Disease-Prediction — hea
eu-de.dataplatform.cloud.ibm.com/ml/auto-ml/6c09992d-620b-42e2-a826-39140f99313b/train?projectid=1ad38531-b4e3-42d6-8ea2-3cc1a5b37d19&context=cpdaas
IBM Cloud Pak for Data
Search in your workspaces
Upgrade
Nigharika S's Account
Frankfurt
NS
Projects / heart-disease-pred / Heart-Disease-Prediction
Experiment summary
Pipeline comparison
★ Rank by: Accuracy (Optimized) | Cross validation score
Relationship map
Prediction column: condition
FEATURE TRANSFORMERS
PIPELINES
TOP ALGORITHMS
heart-disease-dat...
Progress map
Swap view
View log
Save code

Experiment completed

12 PIPELINES GENERATED

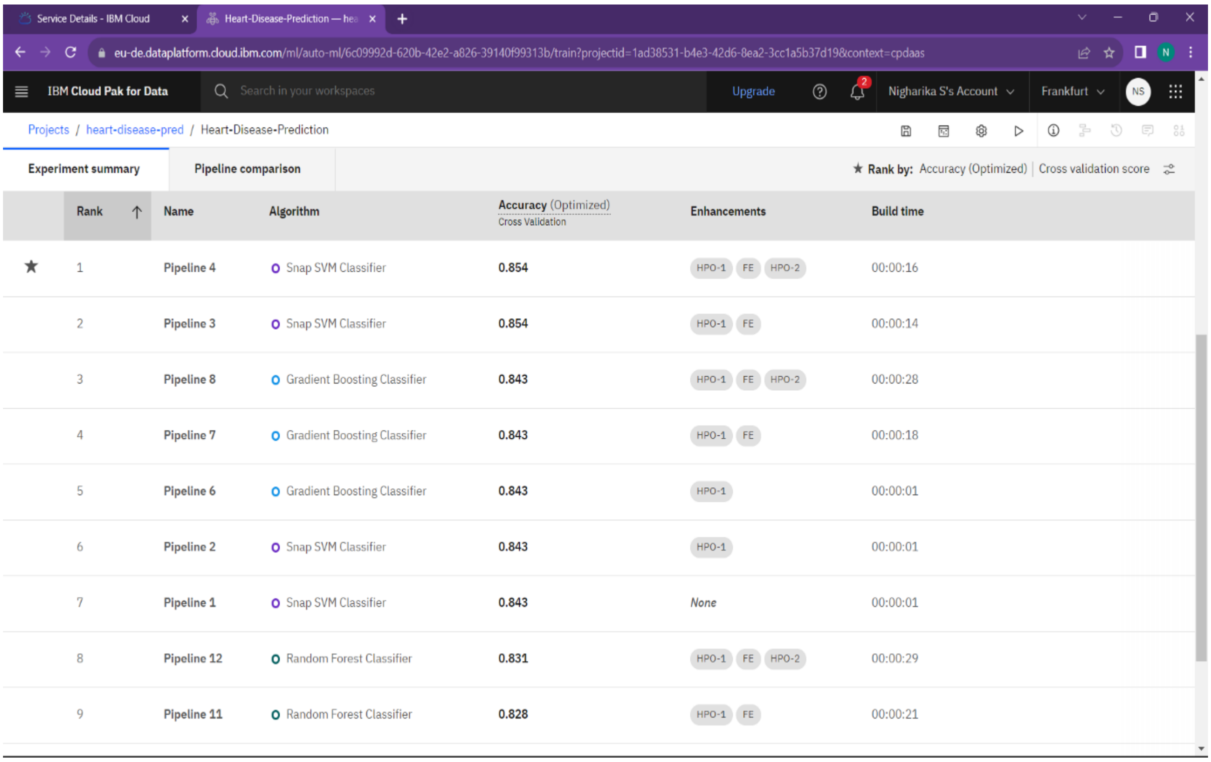
12 pipelines generated from algorithms. See pipeline leaderboard below for more detail.

Time elapsed: 2 minutes

Pipeline leaderboard

Rank	Name	Algorithm	Accuracy (Optimized) Cross Validation	Enhancements	Build time

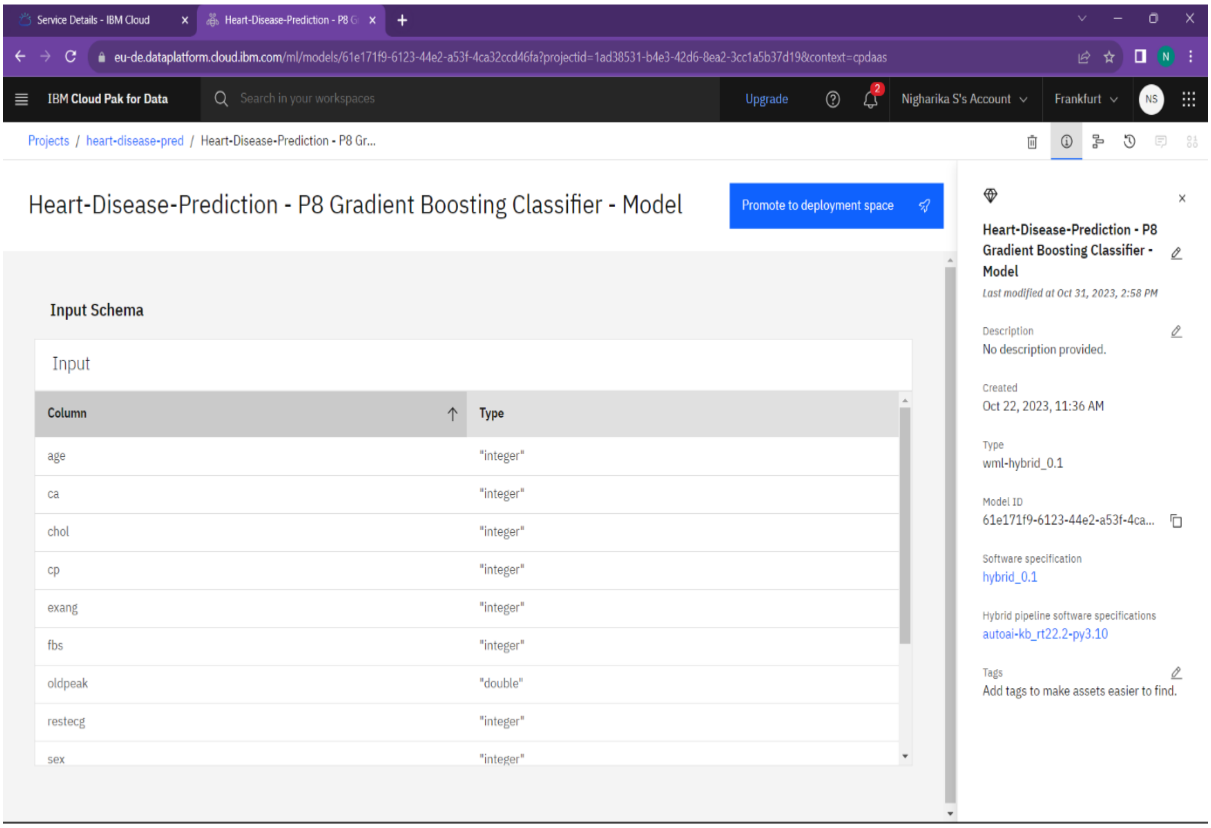
Step 4 : Choosing the Algorithm



Service Details - IBM Cloud
Heart-Disease-Prediction — hea
eu-de.dataplatform.cloud.ibm.com/ml/auto-ml/6c09992d-620b-42e2-a826-39140f99313b/train?projectid=1ad38531-b4e3-42d6-8ea2-3cc1a5b37d19&context=cpdaas
IBM Cloud Pak for Data
Search in your workspaces
Upgrade
Nigharika S's Account
Frankfurt
NS
Projects / heart-disease-pred / Heart-Disease-Prediction
Experiment summary
Pipeline comparison
★ Rank by: Accuracy (Optimized) | Cross validation score

	Rank	Name	Algorithm	Accuracy (Optimized) Cross Validation	Enhancements	Build time
★	1	Pipeline 4	Snap SVM Classifier	0.854	HPO-1 FE HPO-2	00:00:16
	2	Pipeline 3	Snap SVM Classifier	0.854	HPO-1 FE	00:00:14
	3	Pipeline 8	Gradient Boosting Classifier	0.843	HPO-1 FE HPO-2	00:00:28
	4	Pipeline 7	Gradient Boosting Classifier	0.843	HPO-1 FE	00:00:18
	5	Pipeline 6	Gradient Boosting Classifier	0.843	HPO-1	00:00:01
	6	Pipeline 2	Snap SVM Classifier	0.843	HPO-1	00:00:01
	7	Pipeline 1	Snap SVM Classifier	0.843	*None*	00:00:01
	8	Pipeline 12	Random Forest Classifier	0.831	HPO-1 FE HPO-2	00:00:29
	9	Pipeline 11	Random Forest Classifier	0.828	HPO-1 FE	00:00:21

Step 5 : ML Model Created Successfully in IBM Watson



Service Details - IBM Cloud
Heart-Disease-Prediction - P8 G
eu-de.dataplatform.cloud.ibm.com/ml/models/61e171f9-6123-44e2-a53f-4ca32ccd46fa?projectid=1ad38531-b4e3-42d6-8ea2-3cc1a5b37d19&context=cpdaas
IBM Cloud Pak for Data
Search in your workspaces
Upgrade
Nigharika S's Account
Frankfurt
NS
Projects / heart-disease-pred / Heart-Disease-Prediction - P8 Gr...
Promote to deployment space

Heart-Disease-Prediction - P8 Gradient Boosting Classifier - Model

Input Schema

Input

Column	Type
age	"integer"
ca	"integer"
chol	"integer"
cp	"integer"
exang	"integer"
fbs	"integer"
oldpeak	"double"
restecg	"integer"
sex	"integer"

Heart-Disease-Prediction - P8 Gradient Boosting Classifier - Model

Last modified at Oct 31, 2023, 2:58 PM

Description
No description provided.

Created
Oct 22, 2023, 11:36 AM

Type
wml-hybrid_0.1

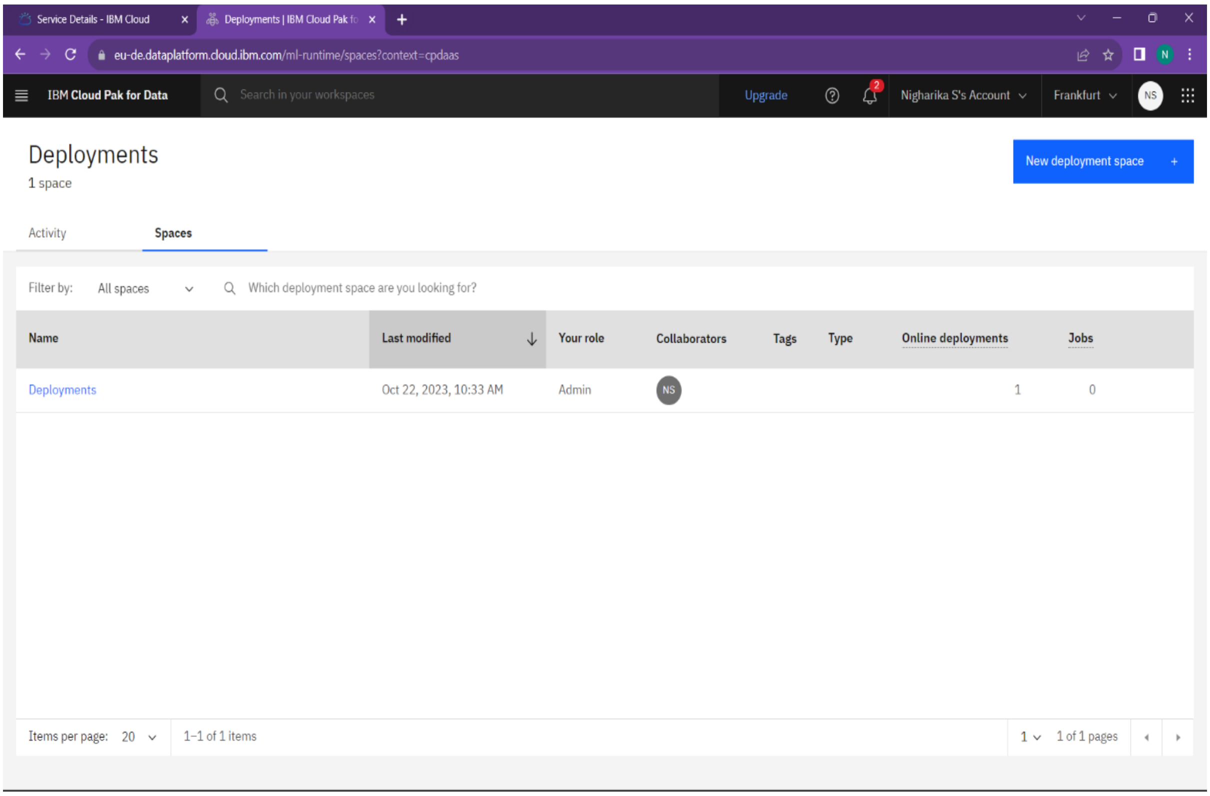
Model ID
61e171f9-6123-44e2-a53f-4ca...

Software specification
hybrid_0.1

Hybrid pipeline software specifications
autoai-kb_rt22.2-py3.10

Tags
Add tags to make assets easier to find.

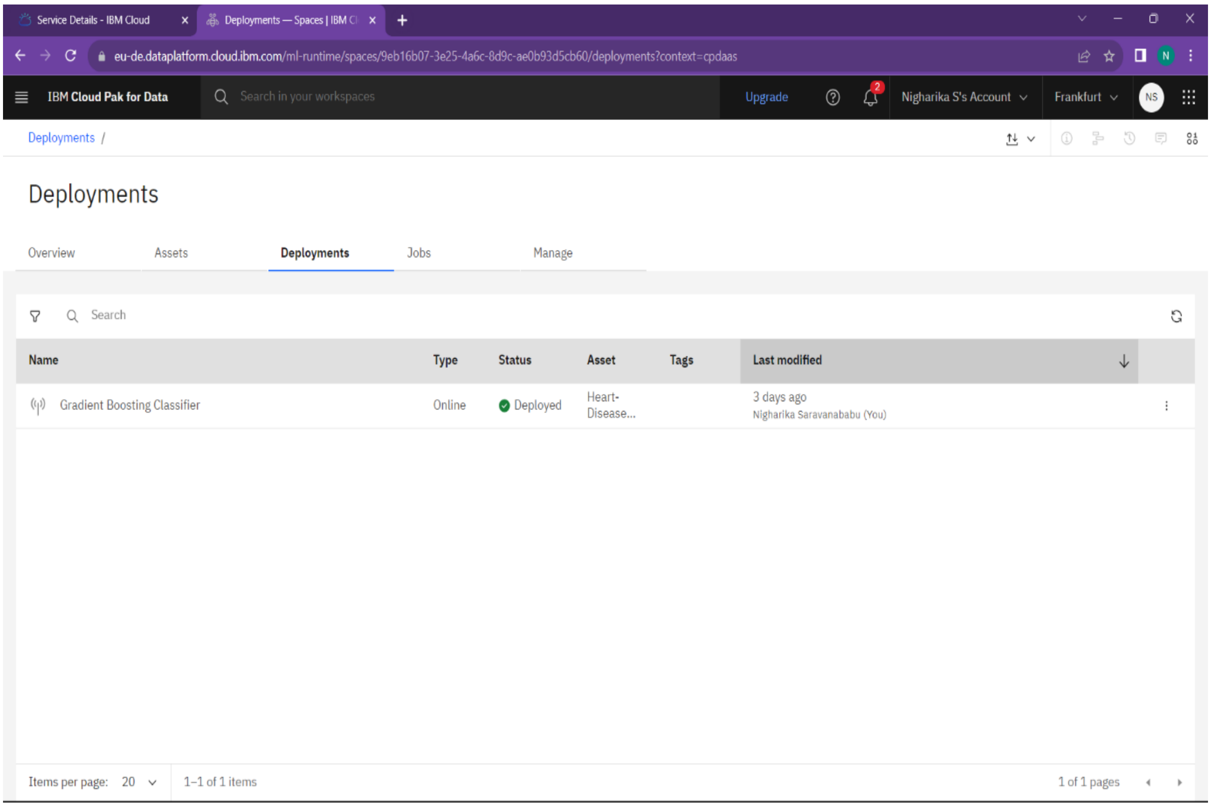
Step 6: Create a new deployment space and Deploy the model as web service


Service Details - IBM Cloud
Deployments | IBM Cloud Pak fo
eu-de.dataplatform.cloud.ibm.com/ml-runtime/spaces?context=cpdaas
IBM Cloud Pak for Data
Search in your workspaces
Upgrade
Nigharika S's Account
Frankfurt
NS
New deployment space +
Activity
Spaces
Filter by: All spaces
Which deployment space are you looking for?
Items per page: 20
1–1 of 1 items
1 of 1 pages

Deployments

1 space

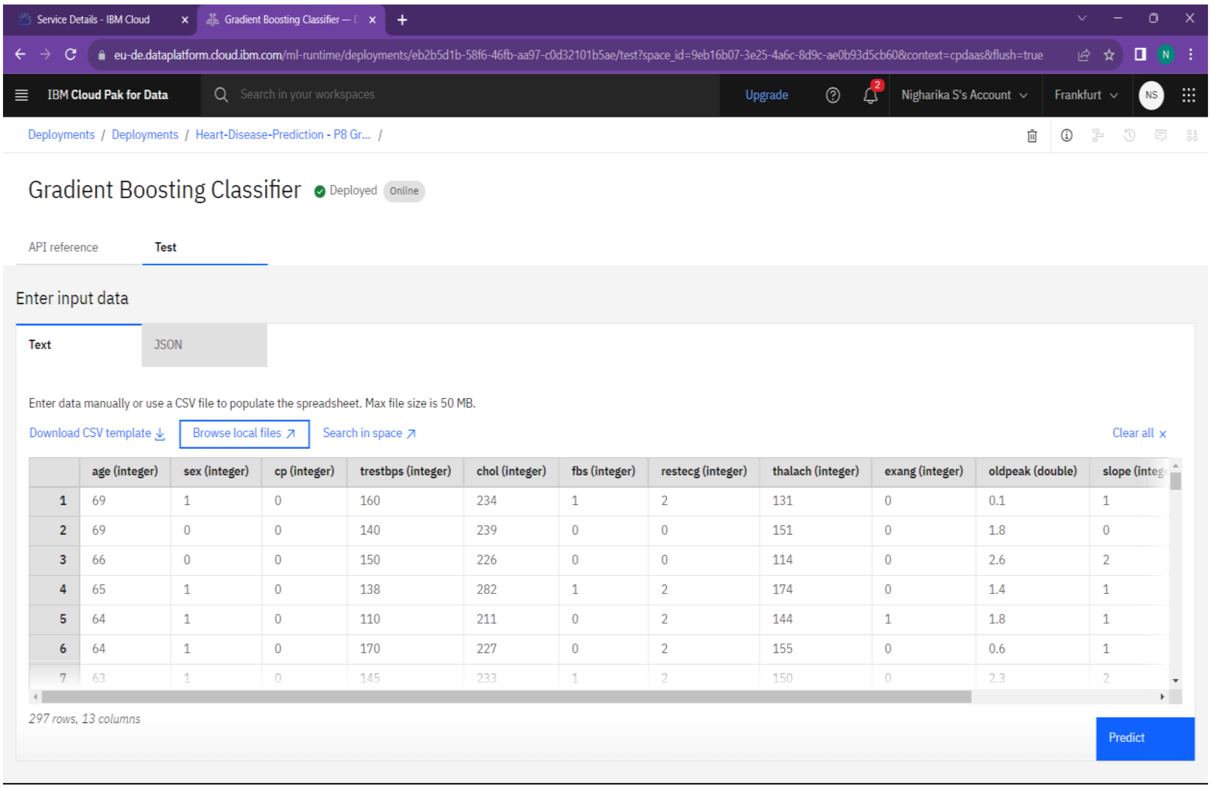
Name	Last modified	Your role	Collaborators	Tags	Type	Online deployments	Jobs
Deployments	Oct 22, 2023, 10:33 AM	Admin	NS			1	0


Service Details - IBM Cloud
Deployments — Spaces | IBM Cl
eu-de.dataplatform.cloud.ibm.com/ml-runtime/spaces/9eb16b07-3e25-4a6c-8d9c-ae0b93d5cb60/deployments?context=cpdaas
IBM Cloud Pak for Data
Search in your workspaces
Upgrade
Nigharika S's Account
Frankfurt
NS
Deployments /
Overview
Assets
Deployments
Jobs
Manage
Search
Items per page: 20
1–1 of 1 items
1 of 1 pages

Deployments

Name	Type	Status	Asset	Tags	Last modified
Gradient Boosting Classifier	Online	● Deployed	Heart-Disease...		3 days ago Nigharika Saravanababu (You)

Step 7 : Test the Model by importing the data set (without prediction column) by clicking Browse Local Files



Service Details - IBM Cloud | Gradient Boosting Classifier — I

eu-de.dataplatform.cloud.ibm.com/ml-runtime/deployments/eb2b5d1b-58f6-46fb-aa97-c0d32101b5ae/test?space_id=9eb16b07-3e25-4a6c-8d9c-ae0b93d5cb60&context=cpdaas&flush=true

IBM Cloud Pak for Data | Search in your workspaces | Upgrade | Nigharika S's Account | Frankfurt

Deployments / Deployments / Heart-Disease-Prediction - P8 Gr... /

Gradient Boosting Classifier ● Deployed Online

API reference | Test

Enter input data

Text | JSON

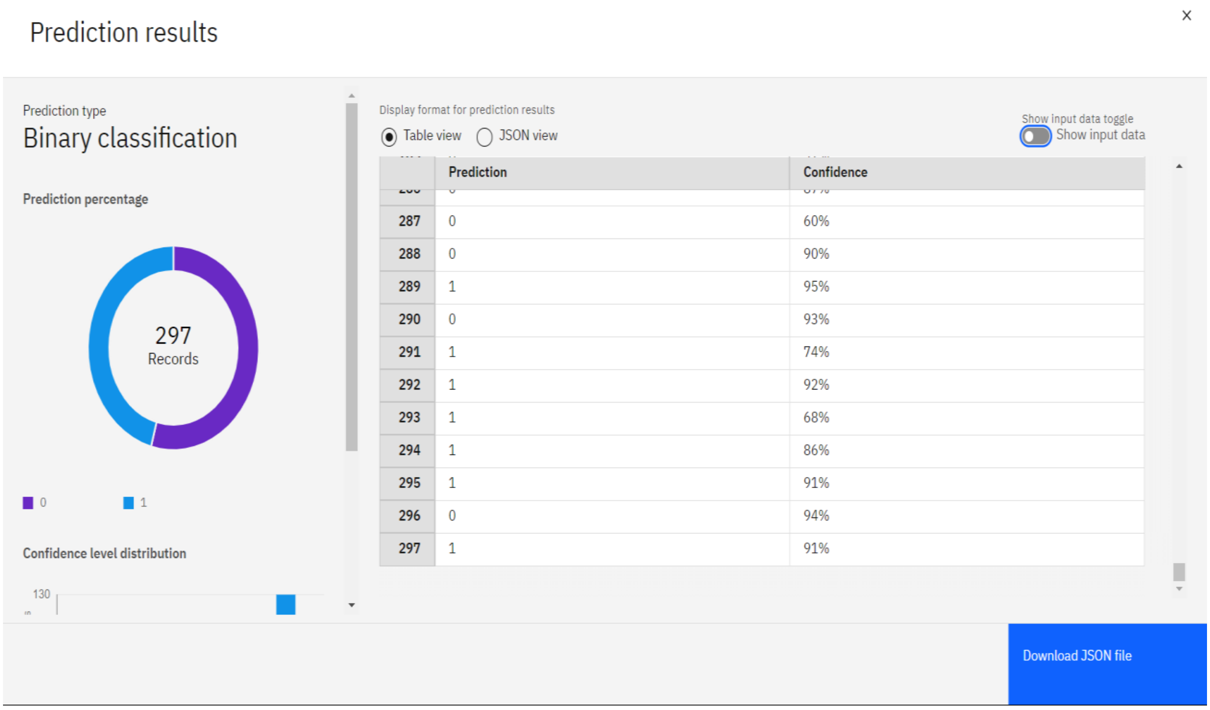
Enter data manually or use a CSV file to populate the spreadsheet. Max file size is 50 MB.

Download CSV template | Browse local files | Search in space | Clear all ×

	age (integer)	sex (integer)	cp (integer)	trestbps (integer)	chol (integer)	fbs (integer)	restecg (integer)	thalach (integer)	exang (integer)	oldpeak (double)	slope (integ
1	69	1	0	160	234	1	2	131	0	0.1	1
2	69	0	0	140	239	0	0	151	0	1.8	0
3	66	0	0	150	226	0	0	114	0	2.6	2
4	65	1	0	138	282	1	2	174	0	1.4	1
5	64	1	0	110	211	0	2	144	1	1.8	1
6	64	1	0	170	227	0	2	155	0	0.6	1
7	63	1	0	145	233	1	2	150	0	2.3	2

297 rows, 13 columns

Predict



Prediction results

Prediction type

Binary classification

Prediction percentage

Confidence level distribution

Display format for prediction results

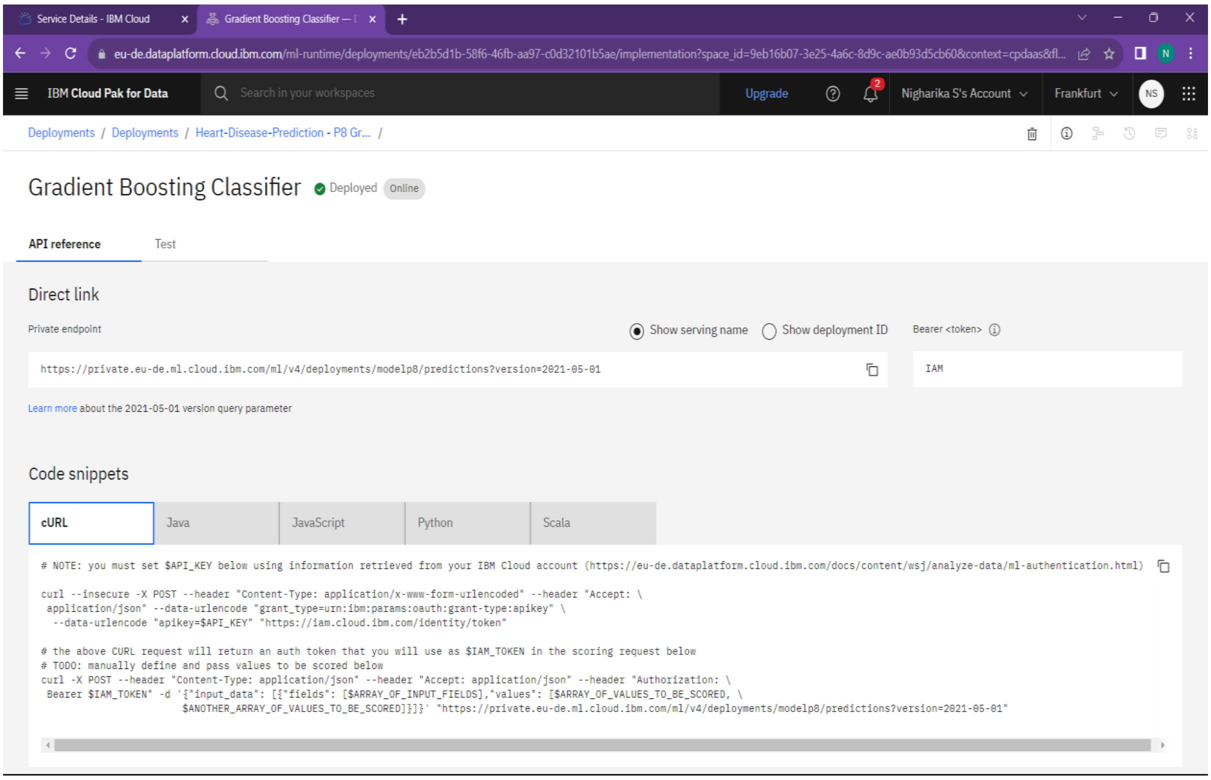
● Table view ○ JSON view

Show input data toggle: Show input data

	Prediction	Confidence
287	0	60%
288	0	90%
289	1	95%
290	0	93%
291	1	74%
292	1	92%
293	1	68%
294	1	86%
295	1	91%
296	0	94%
297	1	91%

Download JSON file

Step 8 : Go to Deployment and get the API REFERENCE


Service Details - IBM Cloud
Gradient Boosting Classifier — I
eu-de.dataplatform.cloud.ibm.com/ml-runtime/deployments/eb2b5d1b-58f6-46fb-aa97-c0d32101b5ae/implementation?space_id=9eb16b07-3e25-4a6c-8d9c-ae0b93d5cb60&context=cpdaas&fl...
IBM Cloud Pak for Data
Search in your workspaces
Upgrade
Nigharika S's Account
Frankfurt
NS
Deployments / Deployments / Heart-Disease-Prediction - P8 Gr... /

Gradient Boosting Classifier ● Deployed Online

API reference Test

Direct link

Private endpoint ● Show serving name ○ Show deployment ID

```
https://private.eu-de.ml.cloud.ibm.com/ml/v4/deployments/modelp8/predictions?version=2021-05-01
```

Bearer <token>

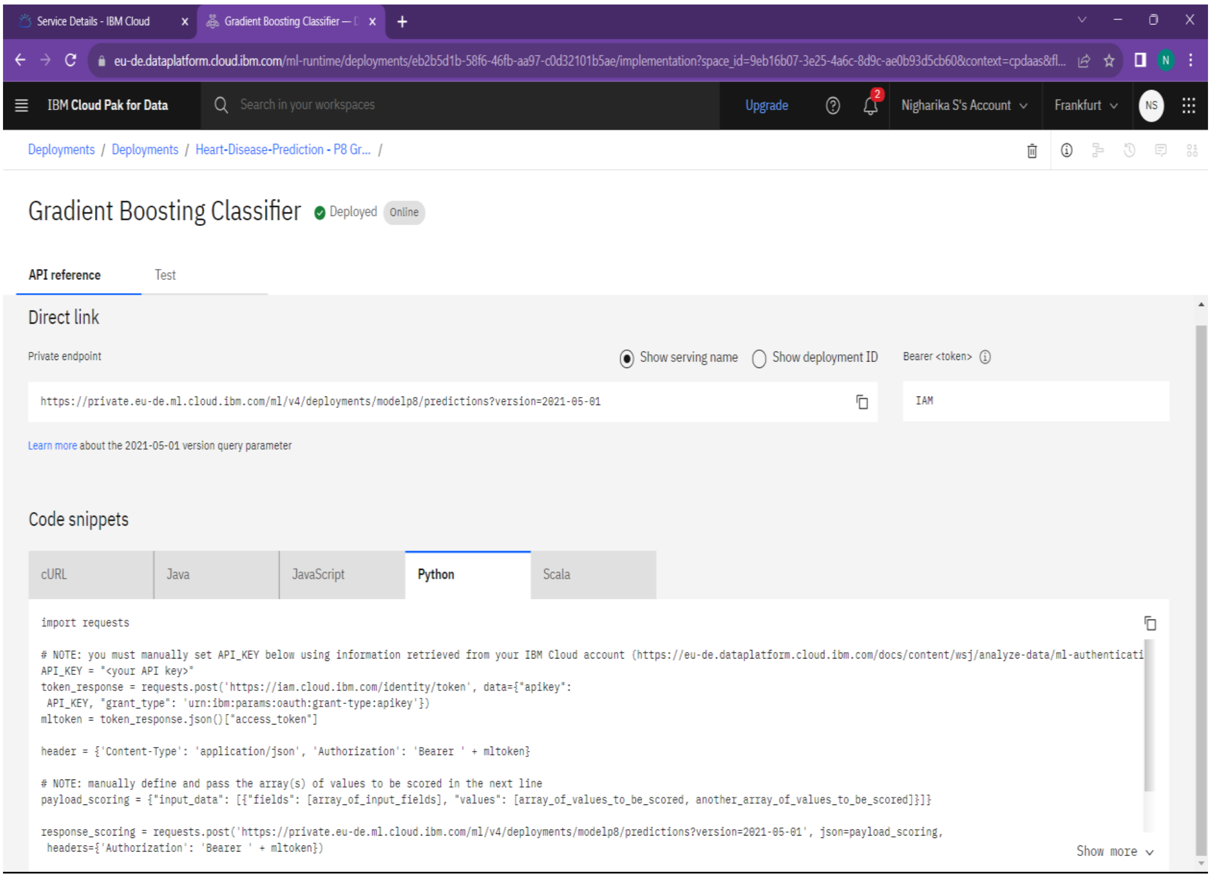
IAM

Learn more about the 2021-05-01 version query parameter

Code snippets

cURL Java JavaScript Python Scala

```
# NOTE: you must set $API_KEY below using information retrieved from your IBM Cloud account (https://eu-de.dataplatform.cloud.ibm.com/docs/content/wsj/analyze-data/ml-authentication.html)

curl --insecure -X POST --header "Content-Type: application/x-www-form-urlencoded" --header "Accept: \
 application/json" --data-urlencode "grant_type=urn:ibm:params:oauth:grant-type:apikey" \
  --data-urlencode "apikey=$API_KEY" "https://iam.cloud.ibm.com/identity/token"

# the above CURL request will return an auth token that you will use as $IAM_TOKEN in the scoring request below
# TODO: manually define and pass values to be scored below
curl -X POST --header "Content-Type: application/json" --header "Accept: application/json" --header "Authorization: \
 Bearer $IAM_TOKEN" -d '{"input_data": [{"fields": [$ARRAY_OF_INPUT_FIELDS],"values": [$ARRAY_OF_VALUES_TO_BE_SCORED, \
                        $ANOTHER_ARRAY_OF_VALUES_TO_BE_SCORED]}]}' "https://private.eu-de.ml.cloud.ibm.com/ml/v4/deployments/modelp8/predictions?version=2021-05-01"
```


Service Details - IBM Cloud
Gradient Boosting Classifier — I
eu-de.dataplatform.cloud.ibm.com/ml-runtime/deployments/eb2b5d1b-58f6-46fb-aa97-c0d32101b5ae/implementation?space_id=9eb16b07-3e25-4a6c-8d9c-ae0b93d5cb60&context=cpdaas&fl...
IBM Cloud Pak for Data
Search in your workspaces
Upgrade
Nigharika S's Account
Frankfurt
NS
Deployments / Deployments / Heart-Disease-Prediction - P8 Gr... /

Gradient Boosting Classifier ● Deployed Online

API reference Test

Direct link

Private endpoint ● Show serving name ○ Show deployment ID

```
https://private.eu-de.ml.cloud.ibm.com/ml/v4/deployments/modelp8/predictions?version=2021-05-01
```

Bearer <token>

IAM

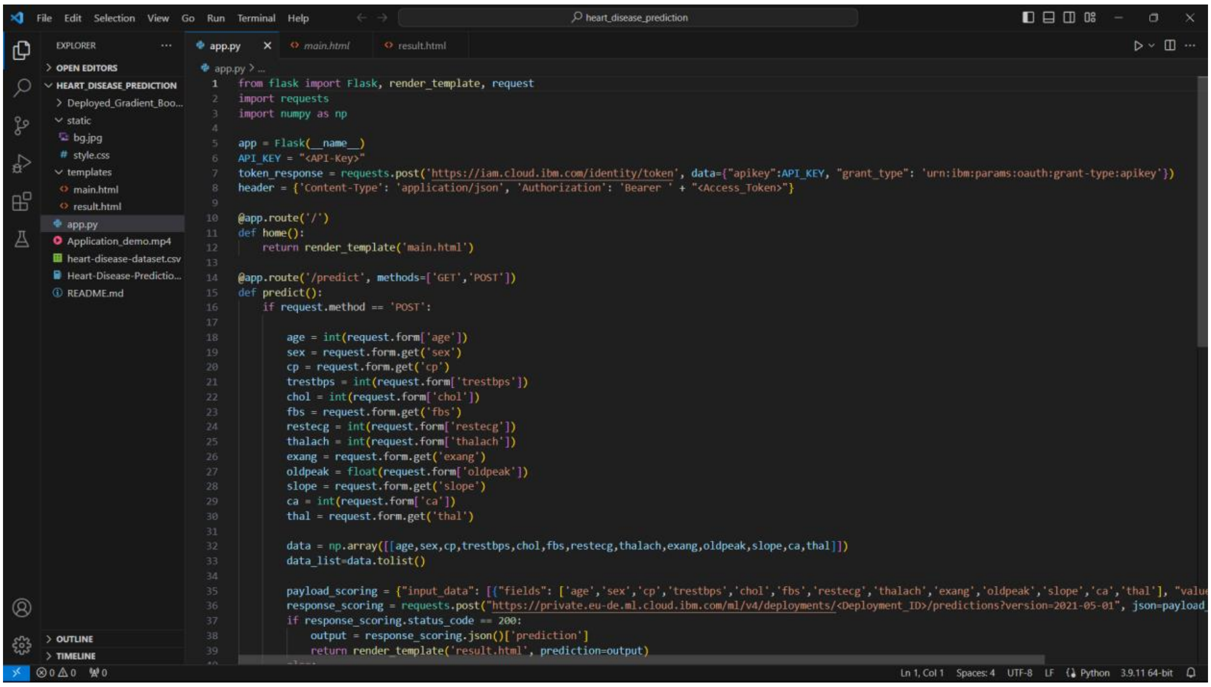
Learn more about the 2021-05-01 version query parameter

Code snippets

cURL Java JavaScript **Python** Scala

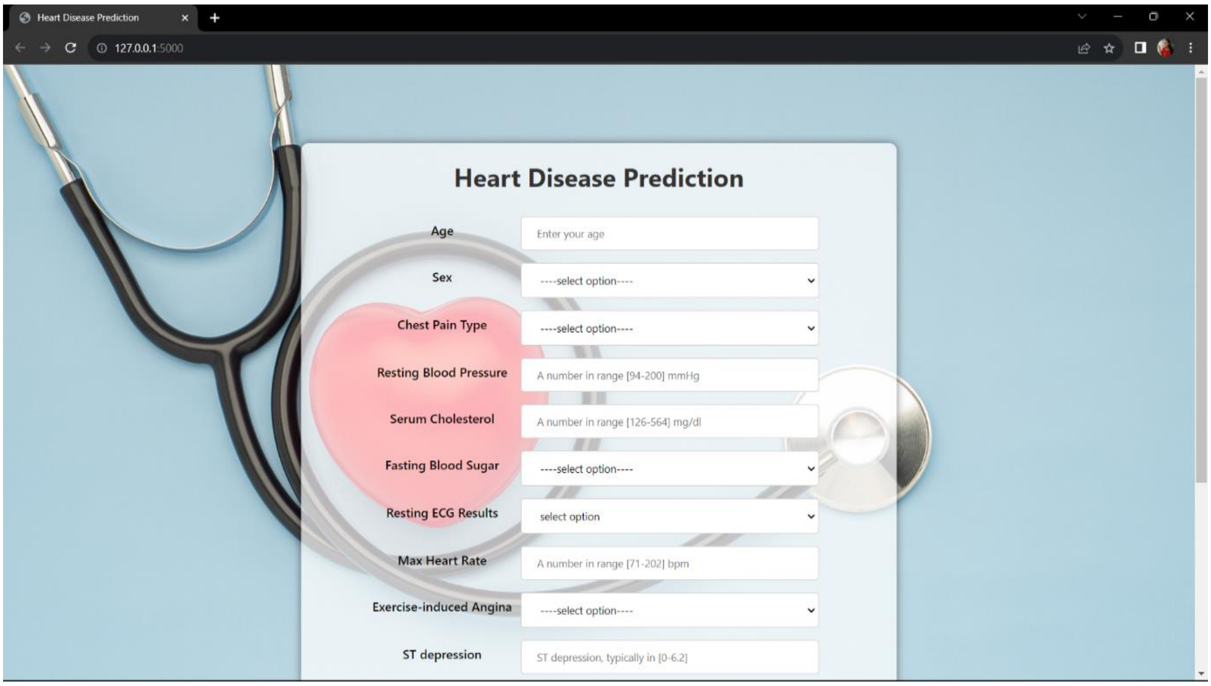
```
import requests

# NOTE: you must manually set API_KEY below using information retrieved from your IBM Cloud account (https://eu-de.dataplatform.cloud.ibm.com/docs/content/wsj/analyze-data/ml-authenticati
API_KEY = "<your API key>"
token_response = requests.post('https://iam.cloud.ibm.com/identity/token', data={"apikey":
 API_KEY, "grant_type": 'urn:ibm:params:oauth:grant-type:apikey'})
mltoken = token_response.json()["access_token"]

header = {'Content-Type': 'application/json', 'Authorization': 'Bearer ' + mltoken}

# NOTE: manually define and pass the array(s) of values to be scored in the next line
payload_scoring = {"input_data": [{"fields": [array_of_input_fields], "values": [array_of_values_to_be_scored, another_array_of_values_to_be_scored]}]}

response_scoring = requests.post('https://private.eu-de.ml.cloud.ibm.com/ml/v4/deployments/modelp8/predictions?version=2021-05-01', json=payload_scoring,
 headers={'Authorization': 'Bearer ' + mltoken})
```

Step 9 : By using API Access the Model



```python
from flask import Flask, render_template, request
import requests
import numpy as np

app = Flask(__name__)
API_KEY = "<API-Key>"
token_response = requests.post('https://iam.cloud.ibm.com/identity/token', data={"apikey":API_KEY, "grant_type": 'urn:ibm:params:oauth:grant-type:apikey'})
header = {'Content-Type': 'application/json', 'Authorization': 'Bearer ' + "<Access_Token>"}

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

@app.route('/predict', methods=['GET','POST'])
def predict():
    if request.method == 'POST':

        age = int(request.form['age'])
        sex = request.form.get('sex')
        cp = request.form.get('cp')
        trestbps = int(request.form['trestbps'])
        chol = int(request.form['chol'])
        fbs = request.form.get('fbs')
        restecg = int(request.form['restecg'])
        thalach = int(request.form['thalach'])
        exang = request.form.get('exang')
        oldpeak = float(request.form['oldpeak'])
        slope = request.form.get('slope')
        ca = int(request.form['ca'])
        thal = request.form.get('thal')

        data = np.array([[age,sex,cp,trestbps,chol,fbs,restecg,thalach,exang,oldpeak,slope,ca,thal]])
        data_list=data.tolist()

        payload_scoring = {"input_data": [{"fields": ['age','sex','cp','trestbps','chol','fbs','restecg','thalach','exang','oldpeak','slope','ca','thal'], "value
        response_scoring = requests.post("https://private.eu-de.ml.cloud.ibm.com/ml/v4/deployments/<Deployment_ID>/predictions?version=2021-05-01", json=payload_
        if response_scoring.status_code == 200:
            output = response_scoring.json()['prediction']
            return render_template('result.html', prediction=output)
```

Step 10 : Give the input data via the user-friendly form



Heart Disease Prediction

Age: Enter your age

Sex: ----select option----

Chest Pain Type: ----select option----

Resting Blood Pressure: A number in range [94-200] mmHg

Serum Cholesterol: A number in range [126-564] mg/dl

Fasting Blood Sugar: ----select option----

Resting ECG Results: select option

Max Heart Rate: A number in range [71-202] bpm

Exercise-induced Angina: ----select option----

ST depression: ST depression, typically in [0-6.2]

API-Reference-Link:

https://private.eu-de.ml.cloud.ibm.com/ml/v4/deployments/eb2b5d1b-58f6-46fb-aa97-c0d32101b5ae/predictions?version=2021-05-01

Api key : SZZcii7WTvayJWGBp9cYt8WOkDvRmMT1pGgnr8jkcYhY

Deployment Instructions:

1. Create a Deployment Space: In your IBM Watson Studio environment, create a deployment space if you haven't already. This space will house the deployed model.

2. Deploy the Model: After creating a deployment space, deploy the machine learning model to that space. Ensure you have selected the model that you want to deploy.

3. Access the API Endpoint: Once the model is successfully deployed, you'll receive an API endpoint URL. This URL is the endpoint for making predictions using the deployed model.

Using the API Endpoint for Predictions:

You can use Python with the `requests` library to make API requests to the endpoint for predictions. Here's an example of how to do this:

```
import requests
API_KEY = "<API-Key>"
```

# Request an access token

# Prepare headers with the access token

```
token_response =
requests.post('https://iam.cloud.ibm.com/identity/token',
data={"apikey":API_KEY, "grant_type": 'urn:ibm:params:oauth:grant-
type:apikey'})
header = {'Content-Type': 'application/json', 'Authorization':
'Bearer ' + "<Access_Token>"}
```

# Define the API endpoint URL

```
 https://private.eu-de.ml.cloud.ibm.com/ml/v4/deployments/ eb2b5d1b-
58f6-46fb-aa97-c0d32101b5ae/predictions?version=2021-05-01
```

#Get the input data from the user via a user-friendly form created using HTML and CSS in index.html

# pass the input to the prediction model in the expected format

```
age = int(request.form['age'])
sex = request.form.get('sex')
cp = request.form.get('cp')
trestbps = int(request.form['trestbps'])
chol = int(request.form['chol'])
fbs = request.form.get('fbs')
restecg = int(request.form['restecg'])
thalach = int(request.form['thalach'])
exang = request.form.get('exang')
oldpeak = float(request.form['oldpeak'])
slope = request.form.get('slope')
ca = int(request.form['ca'])
thal = request.form.get('thal')

data =
np.array([[age,sex,cp,trestbps,chol,fbs,restecg,thalach,exang,oldpea
k,slope,ca,thal]])
data_list=data.tolist()

payload_scoring = {"input_data": [{"fields":
['age','sex','cp','trestbps','chol','fbs','restecg','thalach','exang
','oldpeak','slope','ca','thal'], "values": [data_list]}]}
```

# Make a POST request to the API endpoint for predictions

```
response_scoring = requests.post("https://private.eu-
de.ml.cloud.ibm.com/ml/v4/deployments/eb2b5d1b-58f6-46fb-aa97-
c0d32101b5ae /predictions?version=2021-05-01", json=payload_scoring,
headers=header)
```

# send the prediction response to the result.html

```
 return render_template('result.html', prediction=output)
```

11.Conclusion:

Our project successfully achieved its objectives, culminating in the deployment of the most accurate heart disease prediction model, Gradient Boosting, on IBM Cloud Watson Studio. The model is now accessible through a user-friendly Flask application, making it easier for users to make informed healthcare decisions. This project represents a significant step in leveraging machine learning for the benefit of healthcare and highlights the potential of deploying models in cloud environments for widespread use.