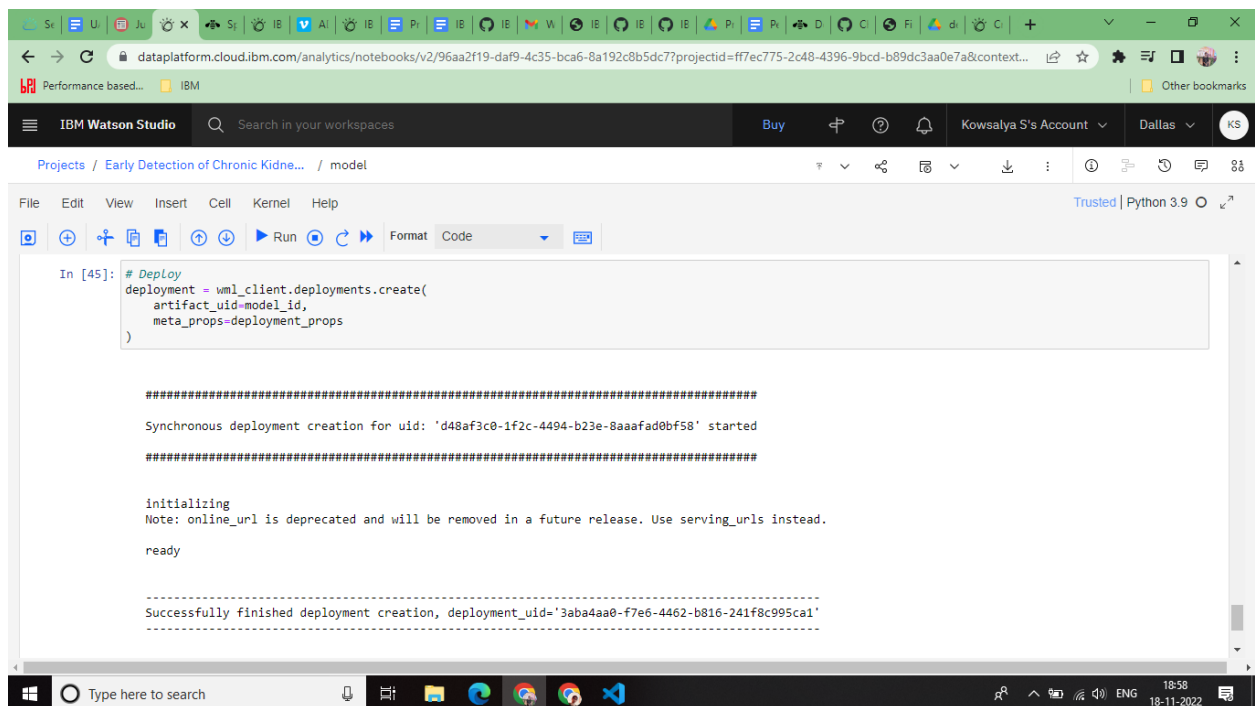


IBM DEPLOYMENT

Date	19 November 2022
Team ID	PNT2022TMID04381
Project Name	Project - Early Detection of Chronic Kidney Disease Using Machine Learning

DEPLOYMENT:



The screenshot displays the IBM Watson Studio web interface. The browser address bar shows a URL from datapatform.cloud.ibm.com. The page header includes the IBM Watson Studio logo and a search bar. The main content area shows a Jupyter Notebook with a code cell. The code cell contains a deployment command and its output. The output shows a successful deployment creation with a deployment_id.

```
In [45]: # Deploy
deployment = wml_client.deployments.create(
    artifact_uid=model_id,
    meta_props=deployment_props
)

=====

Synchronous deployment creation for uid: 'd48af3c0-1f2c-4494-b23e-8aaafad0bf58' started

=====

initializing
Note: online_url is deprecated and will be removed in a future release. Use serving_urls instead.

ready

-----
Successfully finished deployment creation, deployment_uid='3aba4aa0-f7e6-4462-b816-241f8c995ca1'
```

Deployments /

model_ibm

Overview Assets Deployments Jobs Manage

Find assets

Import assets

2 assets

All assets

2

Asset types

Models

2

Assets

Name	Last modified	
prediction Model	6 minutes ago Service	
iris Model	40 minutes ago Service	

Items per page: 20

1-2 of 2 items

1 of 1 pages

Drop files here or browse for files to upload.

Stay on the page until upload completes. Incomplete uploads are cancelled.

Deployments /

model_ibm

Overview Assets Deployments Jobs Manage

Search

Name	Type	Status	Asset	Last modified	
model_ibm	Online	Deployed	prediction	5 minutes ago Kowsalya S (You)	

Items per page: 20

1-1 of 1 items

1 of 1 pages

Drop files here or browse for files to upload.

Stay on the page until upload completes. Incomplete uploads are cancelled.

Deployments / model_ibm / prediction /

model_ibm

Deployed Online

API reference

Test

Direct link

Endpoint

https://us-south.ml.cloud.ibm.com/ml/v4/deployments/3aba4aa0-f7e6-4462-b816-241f8c995ca1?space_id=a2703323-87ed-47d3-9b8e-73636cf5ea75&content_type=application/json

Bearer <token>

IAM

Code snippets

cURL

Java

JavaScript

Python

Scala

```
# NOTE: you must set $API_KEY below using information retrieved from your IBM Cloud account.

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
```

model_ibm

Created

Nov 18, 2022, 6:52 PM

Updated

Nov 18, 2022, 6:52 PM

Deployment ID

3aba4aa0-f7e6-4462-b816-24...

Software specification

runtime-22.1-py3.9

Copies

1

Serving name

No serving name.

Description

No description provided.

Tags

Deployments / model_ibm / prediction /

model_ibm

Deployed Online

API reference

Test

Enter input data

Text input

JSON input

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

[Download CSV template](#)

[Browse local files](#)

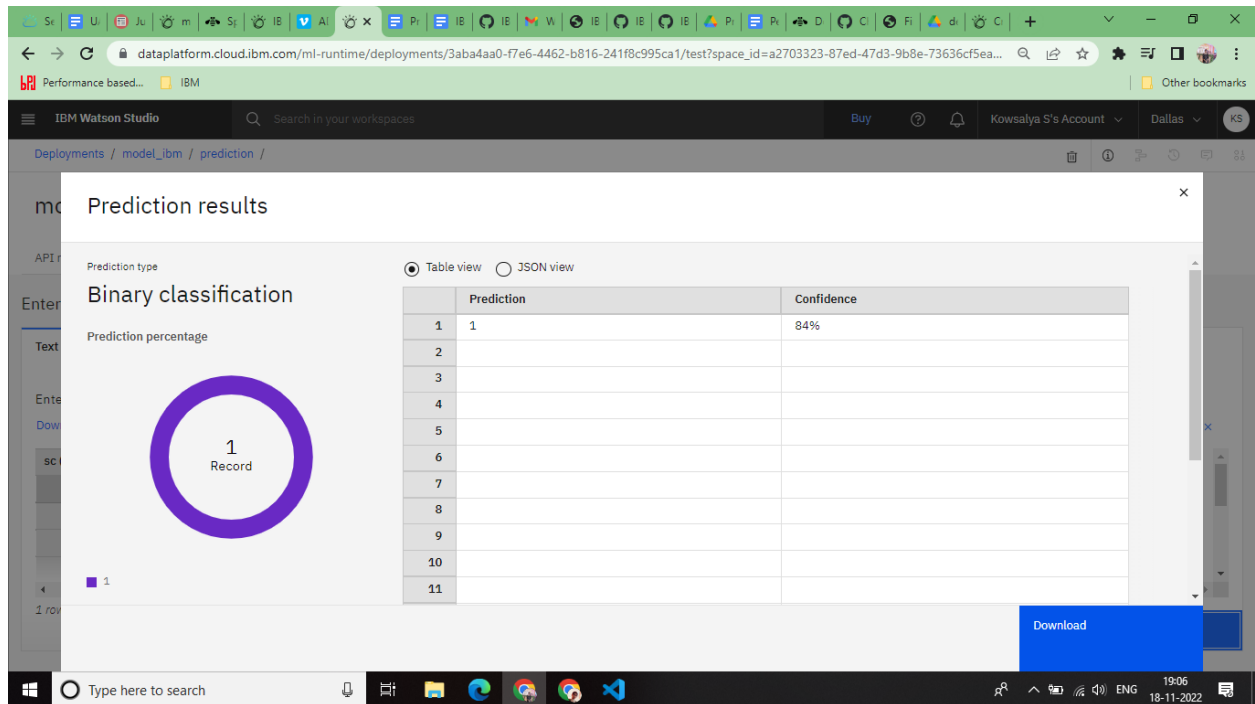
[Search in space](#)

[Clear all](#)

sc (float64)	sod (float64)	pot (float64)	hemo (float64)	wc (float64)	rc (float64)	htn (float64)	dm (float64)	cad (float64)	appet (float64)	pe (float64)	ane (float64)
1	138.0	4.4	15.4	7800.0	5.2	1.0	1.0	0.0	1.0	0.0	0.0
2											
3											
4											

1 row, 23 columns

Predict



MODEL.PY

```
model.py - Early Detection of Chronic Kidney Disease - Visual Studio Code

4 # Importing Libraries:
5 import pandas as pd
6 import numpy as np
7 import pickle
8
9 # for displaying all feature from dataset:
10 pd.pandas.set_option('display.max_columns', None)
11
12 # Reading Dataset:
13 dataset = pd.read_csv("Kidney_data.csv")
14
15 # Dropping unnecessary feature :|
16 dataset = dataset.drop('id', axis=1)
17
18 # Replacing Categorical Values with Numericals
19 dataset['rbc'] = dataset['rbc'].replace(to_replace = {'normal' : 0, 'abnormal' : 1})
20 dataset['pc'] = dataset['pc'].replace(to_replace = {'normal' : 0, 'abnormal' : 1})
21 dataset['pcc'] = dataset['pcc'].replace(to_replace = {'notpresent':0,'present':1})
22 dataset['ba'] = dataset['ba'].replace(to_replace = {'notpresent':0,'present':1})
23 dataset['htn'] = dataset['htn'].replace(to_replace = {'yes' : 1, 'no' : 0})
24
25 dataset['dm'] = dataset['dm'].replace(to_replace = {'\tyes':'yes', ' yes':'yes', '\tno':'no'})
26 dataset['dm'] = dataset['dm'].replace(to_replace = {'yes' : 1, 'no' : 0})
27
28 dataset['cad'] = dataset['cad'].replace(to_replace = {'\tno':'no'})
29 dataset['cad'] = dataset['cad'].replace(to_replace = {'yes' : 1, 'no' : 0})
30
31 dataset['appet'] = dataset['appet'].replace(to_replace={'good':1,'poor':0,'no':np.nan})
32 dataset['pe'] = dataset['pe'].replace(to_replace = {'yes' : 1, 'no' : 0})
33 dataset['ane'] = dataset['ane'].replace(to_replace = {'yes' : 1, 'no' : 0})
34
35 dataset['classification'] = dataset['classification'].replace(to_replace={'ckd\t':'ckd'})
```

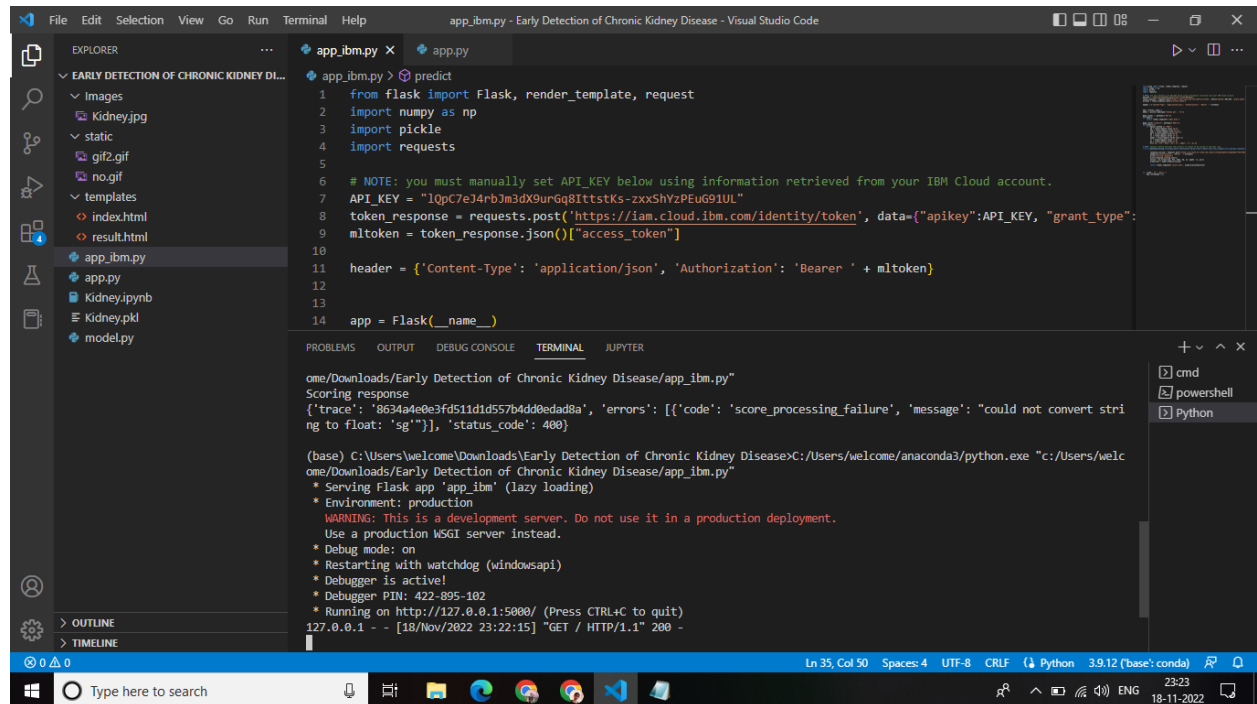
BUILD PYTHON CODE:

```
1 from flask import Flask, render_template, request
2 import numpy as np
3 import pickle
4
5
6 app = Flask(__name__)
7 model = pickle.load(open('Kidney.pkl', 'rb'))
8
9 @app.route('/', methods=['GET'])
10 def Home():
11     return render_template('index.html')
12
13 @app.route("/predict", methods=['POST'])
14 def predict():
15     if request.method == 'POST':
16         sg = float(request.form['sg'])
17         htn = float(request.form['htn'])
18         hemo = float(request.form['hemo'])
19         dm = float(request.form['dm'])
20         al = float(request.form['al'])
21         appet = float(request.form['appet'])
22         rc = float(request.form['rc'])
23         pc = float(request.form['pc'])
24
25         values = np.array([[sg, htn, hemo, dm, al, appet, rc, pc]])
26         prediction = model.predict(values)
27
28         return render_template('result.html', prediction=prediction)
29
30
31 if __name__ == "__main__":
32     app.run(debug=True)
33
```

APP_IBM.PY

```
1 from flask import Flask, render_template, request
2 import numpy as np
3 import pickle
4 import requests
5
6 # NOTE: you must manually set API_KEY below using information retrieved from your IBM Cloud account.
7 API_KEY = "l0pC7eJ4rbJm3dX9urGq8lTtstks-zxxSHYzPEUG91UL"
8 token_response = requests.post("https://iam.cloud.ibm.com/identity/token", data={"apikey":API_KEY, "grant_type":
9 mltoken = token_response.json()["access_token"]
10
11 header = {'Content-Type': 'application/json', 'Authorization': 'Bearer ' + mltoken}
12 app = Flask(__name__)
13 model = pickle.load(open('Kidney.pkl', 'rb'))
14 @app.route('/', methods=['GET'])
15 def Home():
16     return render_template('index.html')
17 @app.route("/predict", methods=['POST'])
18
19 def predict():
20     if request.method == 'POST':
21         sg = float(request.form['sg'])
22         htn = float(request.form['htn'])
23         hemo = float(request.form['hemo'])
24         dm = float(request.form['dm'])
25         al = float(request.form['al'])
26         appet = float(request.form['appet'])
27         rc = float(request.form['rc'])
28         pc = float(request.form['pc'])
29         X=[['sg','htn','hemo','dm','al','appet','rc','pc']]
30
31 # NOTE: manually define and pass the array(s) of values to be scored in the next line
32 payload_scoring = {"input_data": [{"field": [['sg','htn','hemo','dm','al','appet','rc','pc']], "values":
33
```

RUN THE APP:



The screenshot shows the Visual Studio Code interface with the file `app_ibm.py` open. The code is a Flask application for chronic kidney disease prediction. The terminal output shows the application running on `http://127.0.0.1:5000/` and receiving a POST request. The response is a JSON object with a status code of 400 and an error message: `could not convert string to float: 'sg''`.

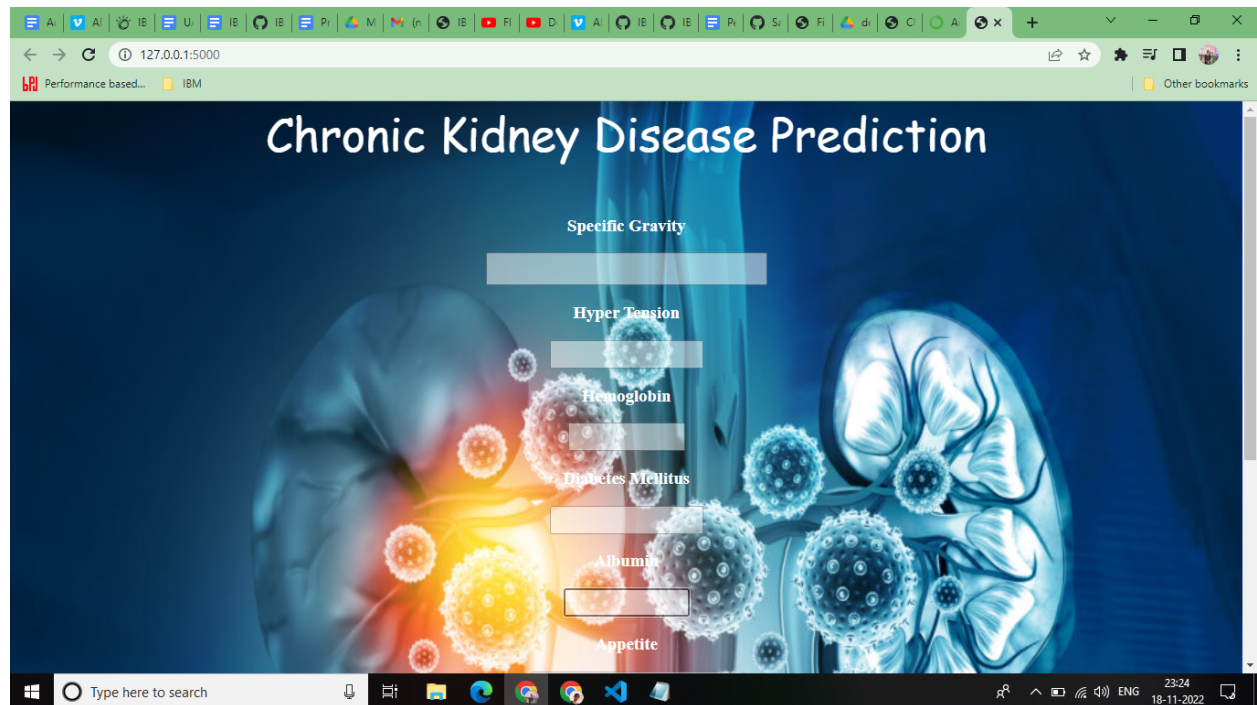
```
app_ibm.py > predict
1 from flask import Flask, render_template, request
2 import numpy as np
3 import pickle
4 import requests
5
6 # NOTE: you must manually set API_KEY below using information retrieved from your IBM Cloud account.
7 API_KEY = "10pC7e14rbJm3dX9urGq8IttstKs-zxxSHYzPEUG01UL"
8 token_response = requests.post("https://iam.cloud.ibm.com/identity/token", data={"apikey":API_KEY, "grant_type":
9 mltoken = token_response.json()["access_token"]
10
11 header = {'Content-Type': 'application/json', 'Authorization': 'Bearer ' + mltoken}
12
13
14 app = Flask(__name__)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER

```
ome/Downloads/Early Detection of Chronic Kidney Disease/app_ibm.py"
Scoring response
{'trace': '8634a4e9e3fd511dd557b4dd0ead8a', 'errors': [{'code': 'score_processing_failure', 'message': "could not convert stri
ng to float: 'sg'"}], 'status_code': 400}

(base) C:\Users\welcome\Downloads\Early Detection of Chronic Kidney Disease>C:\Users\welcome\anaconda3\python.exe "c:/Users/welc
ome/Downloads/Early Detection of Chronic Kidney Disease/app_ibm.py"
* Serving Flask app 'app_ibm' (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 watchdog (windowsapi)
* Debugger is active!
* Debugger PIN: 422-895-102
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
127.0.0.1 - - [18/Nov/2022 23:22:15] "GET / HTTP/1.1" 200 -
```

OUTPUT:




Chronic Kidney Disease Prediction

Oops! 😞

You have CHRONIC KIDNEY DISEASE.

Please Consult Doctor.



Windows taskbar and browser tabs are visible at the bottom of the screenshot.

Chronic Kidney Disease Prediction

🎉 Congratulation! 🎉

You DON'T have Chronic Kidney Disease.



Live a Healthy Life

Windows taskbar and browser tabs are visible at the bottom of the screenshot.