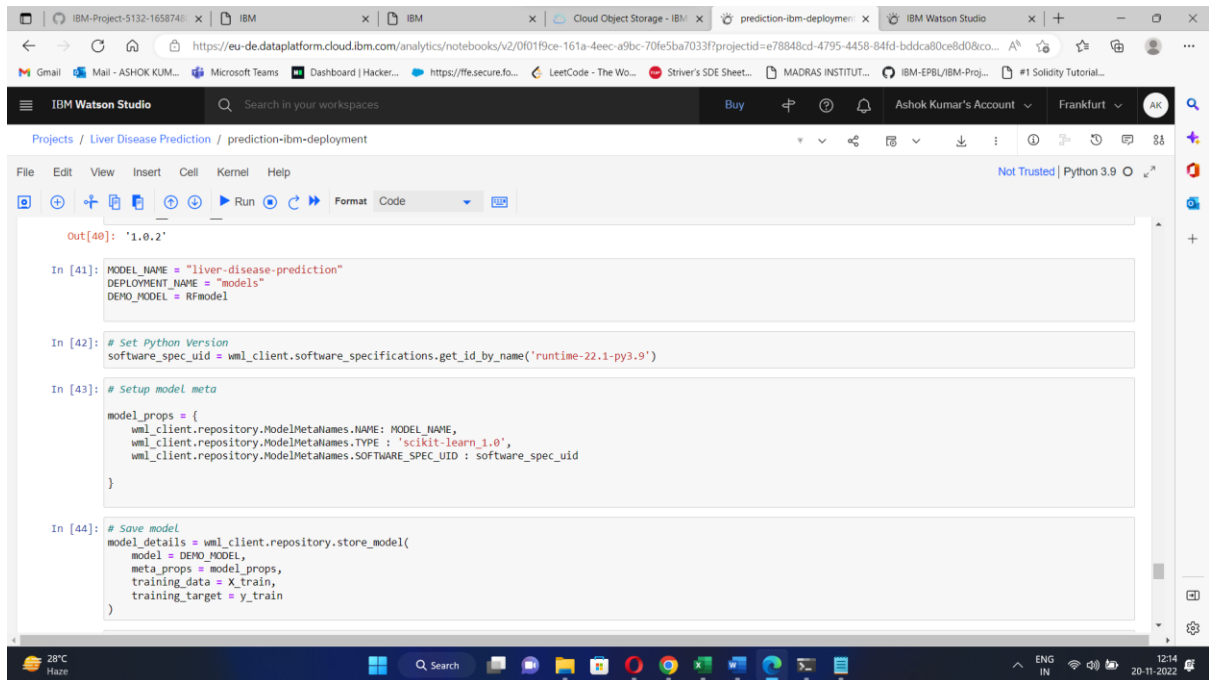


## Training model on IBM Watson:



The screenshot shows the IBM Watson Studio interface with a notebook titled "prediction-ibm-deployment". The notebook contains the following code cells:

```
Out[40]: '1.0.2'
```

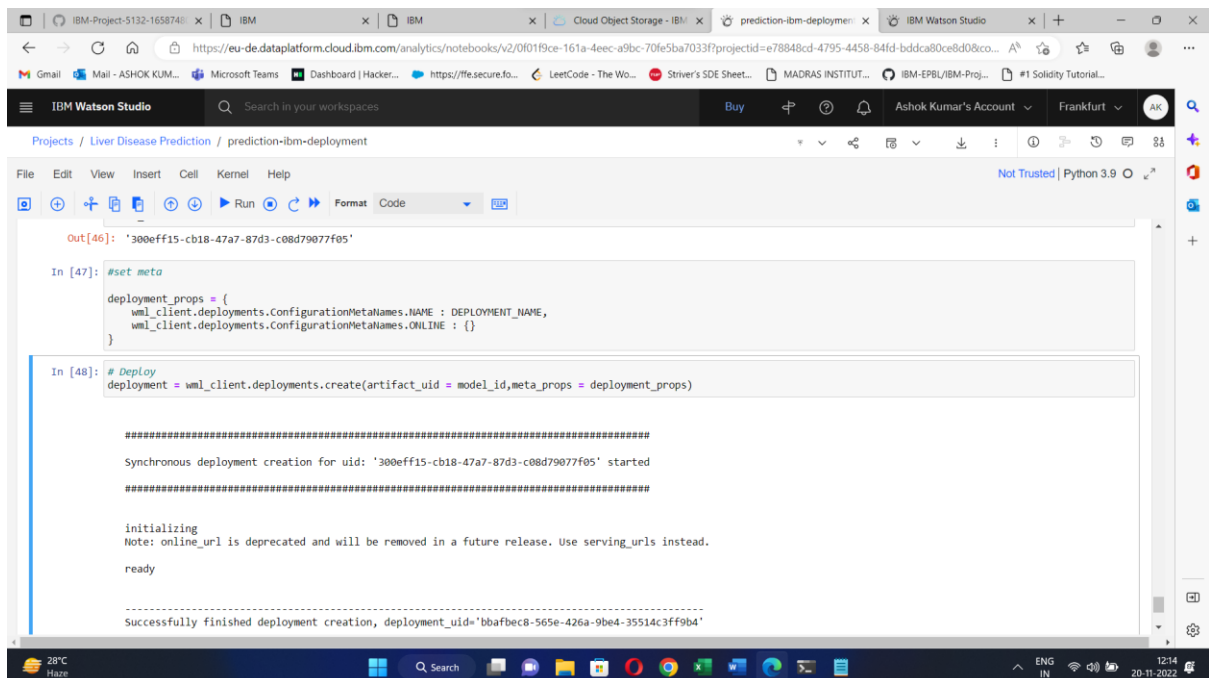
```
In [41]: MODEL_NAME = "liver-disease-prediction"
DEPLOYMENT_NAME = "models"
DEMO_MODEL = RFmodel
```

```
In [42]: # Set Python Version
software_spec_uid = wml_client.software_specifications.get_id_by_name('runtime-22.1-py3.9')
```

```
In [43]: # Setup model meta
model_props = {
    wml_client.repository.ModelMetaNames.NAME: MODEL_NAME,
    wml_client.repository.ModelMetaNames.TYPE: 'scikit-learn-1.0',
    wml_client.repository.ModelMetaNames.SOFTWARE_SPEC_UID: software_spec_uid
}
```

```
In [44]: # Save model
model_details = wml_client.repository.store_model(
    model = DEMO_MODEL,
    meta_props = model_props,
    training_data = X_train,
    training_target = y_train
)
```

## Deploying Model:



The screenshot shows the IBM Watson Studio interface with the same notebook. The code cells for deployment are as follows:

```
Out[46]: '300eff15-cb18-47a7-87d3-c08d79077f05'
```

```
In [47]: #set meta
deployment_props = {
    wml_client.deployments.ConfigurationMetaNames.NAME: DEPLOYMENT_NAME,
    wml_client.deployments.ConfigurationMetaNames.ONLINE: {}
}
```

```
In [48]: # Deploy
deployment = wml_client.deployments.create(artifact_uid = model_id, meta_props = deployment_props)
```

The output of the deployment cell shows the following status:

```
#####
Synchronous deployment creation for uid: '300eff15-cb18-47a7-87d3-c08d79077f05' 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='bbafebcb-565e-426a-9be4-35514c3ff9b4'
```

Deployed model:

The screenshot shows the IBM Watson Studio interface. The main heading is 'liver-disease-prediction'. Below it, there are tabs for 'Deployments' and 'Model details'. The 'Deployments' tab is active, showing a table of deployment types. The table has columns: 'DEPLOYMENT TYPES', 'Name', 'Status', and 'Last modified'. There are two rows: 'Online' (1) and 'Batch' (0). The 'Online' row is expanded, showing a deployment named 'models' with a status of 'Deployed' and a last modified time of 'Nov 20, 2022, 12:13 PM'. A 'New deployment' button is visible in the top right corner of the deployment list.

DEPLOYMENT TYPES	Name	Status	Last modified
Online (1)	models	Deployed	Nov 20, 2022, 12:13 PM
Batch (0)			

The screenshot shows the IBM Watson Studio interface for the 'models' deployment. The main heading is 'models' with a status of 'Deployed' and 'Online'. Below it, there are tabs for 'API reference' and 'Test'. The 'API reference' tab is active, showing a 'Direct link' section with an endpoint URL and a 'Bearer <token>' field. The 'Code snippets' section is also visible, showing a Python code snippet for making a POST request to the endpoint. The right sidebar shows details for the 'models' deployment, including 'Created' (Nov 20, 2022, 12:13 PM), 'Updated' (Nov 20, 2022, 12:13 PM), 'Deployment ID' (bba7bec8-565e-426a-9be4-35514c3ff9b4), 'Software specification' (runtime-22.1-py3.9), 'Copies' (1), 'Serving name' (No serving name), 'Description' (No description provided), 'Tags' (Add tags to make assets easier to find), and 'Associated asset' (liver-disease-prediction).

**Direct link**

Endpoint: <https://eu-de.ml.cloud.ibm.com/v4/deployments/bba7bec8-565e-426a-9be4-35514c3ff9b4/predictions?version=2022-11-15>

Bearer <token>: IAM

**Code snippets**

Python

```
import requests

# NOTE: you must manually set API_KEY below using information retrieved from your IBM Cloud account.
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,
    ...]}]}

response_scoring = requests.post('https://eu-de.ml.cloud.ibm.com/v4/deployments/bba7bec8-565e-426a-9be4-35514c3ff9b4/predictions',
    headers=header, data=payload_scoring)
```