

DEPLOY ON IBM CLOUD OUTPUT

The screenshot displays the IBM Watson Studio interface. At the top, the navigation bar includes the IBM Watson Studio logo, a search bar, and user information (MOTHESHWARA NAVINKU...). The main content area shows the deployment details for 'analysis_deploy_model', which is marked as 'Deployed' and 'Online'. The 'API reference' tab is active, showing the endpoint URL: `https://eu-gb.ml.cloud.ibm.com/ml/v4/deployments/f4a6cc62-cd58-47a3-af62-6a940301a...`. Below the endpoint, there are tabs for 'Code snippets' in different languages: cURL, Java, JavaScript, Python, and Scala. The Python tab is selected, showing a code snippet for making a REST API call to the deployment endpoint. The right sidebar provides additional metadata for the deployment, including its ID, creation and update timestamps, software specification (runtime-22.1-py3.9), and associated asset.

analysis_deploy_model Deployed Online

API reference Test

Endpoint Bearer (token)

`https://eu-gb.ml.cloud.ibm.com/ml/v4/deployments/f4a6cc62-cd58-47a3-af62-6a940301a...` IAM

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.
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

response_scoring = requests.post('https://eu-gb.ml.cloud.ibm.com/ml/v4/deployments/f4a6cc62-cd58-47a3-
headers={'Authorization': 'Bearer ' + mltoken})
```

analysis_deploy_model

Created
Nov 15, 2022, 8:07 PM

Updated
Nov 15, 2022, 8:07 PM

Deployment ID
f4a6cc62-cd58-47a3-af62-6a9...

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.

Associated asset

The screenshot displays the IBM Watson Studio interface for a Jupyter notebook. The notebook is titled 'Importing Libraries' and 'Importing Dataset'. The code in the first cell imports various libraries: pandas, numpy, matplotlib.pyplot, seaborn, and statsmodels.formula.api. The second cell, titled 'Importing Dataset', imports os, types, pandas, boto3, and Config from the boto3 module. It also defines a function to access the IBM Cloud Object Storage bucket and retrieves the 'car performance.csv' file.

Importing Libraries

```
In [1]: import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import statsmodels.formula.api as smf
```

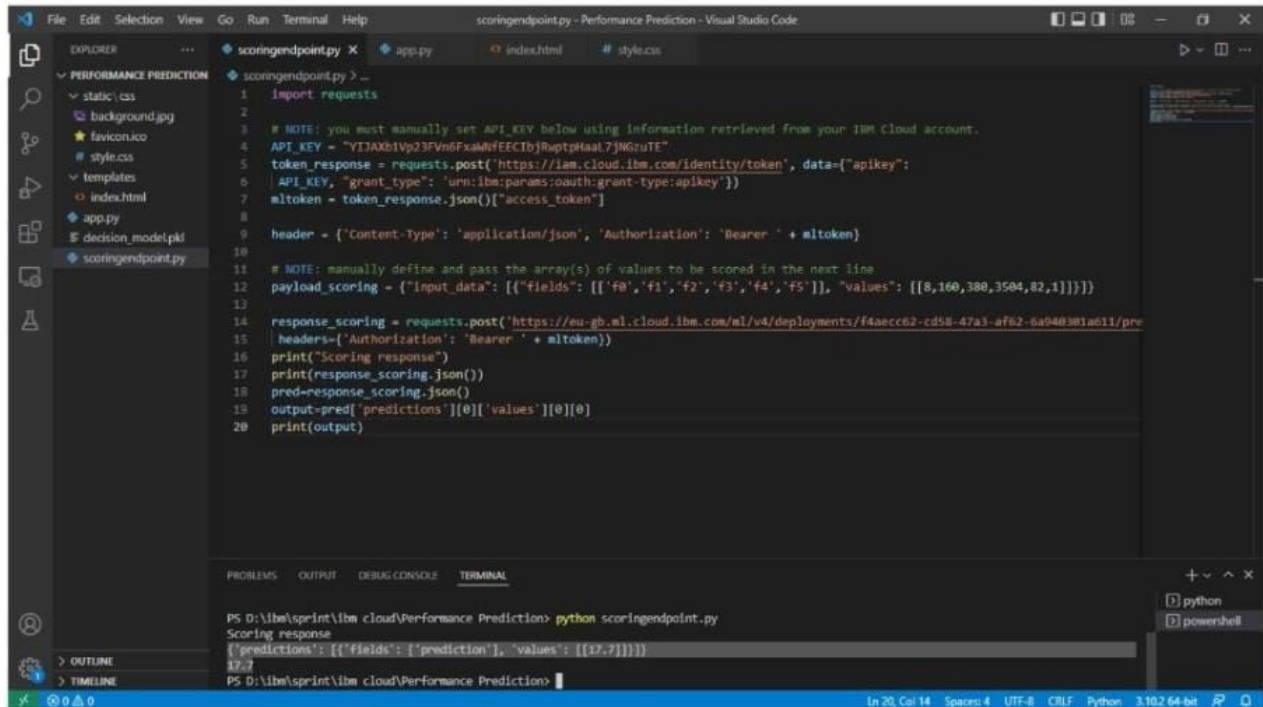
Importing Dataset

```
In [4]: import os, types
import pandas as pd
from boto3.client import Config
import boto3

def __iter__(self): return 0

#@hidden_cell
# The following code accesses a file in your IBM Cloud Object Storage. It includes your credentials.
# You might want to remove those credentials before you share the notebook.
cos_client = boto3.client(service_name='s3',
    ibm_api_key_id='t8Hfza5CCBRth06l3yrkIVT15GkgPZb7U9HQ177ZkE76',
    ibm_auth_endpoint='https://iam.cloud.ibm.com/oidc/token',
    config=Config(signature_version='oauth'),
    endpoint_url='https://s3.private.eu.cloud-object-storage.appdomain.cloud')

bucket = 'machinelearningbasedvehicleperfor-donotdelete-pr-u@oyvtjlsrhov'
object_key = 'car performance.csv'
```



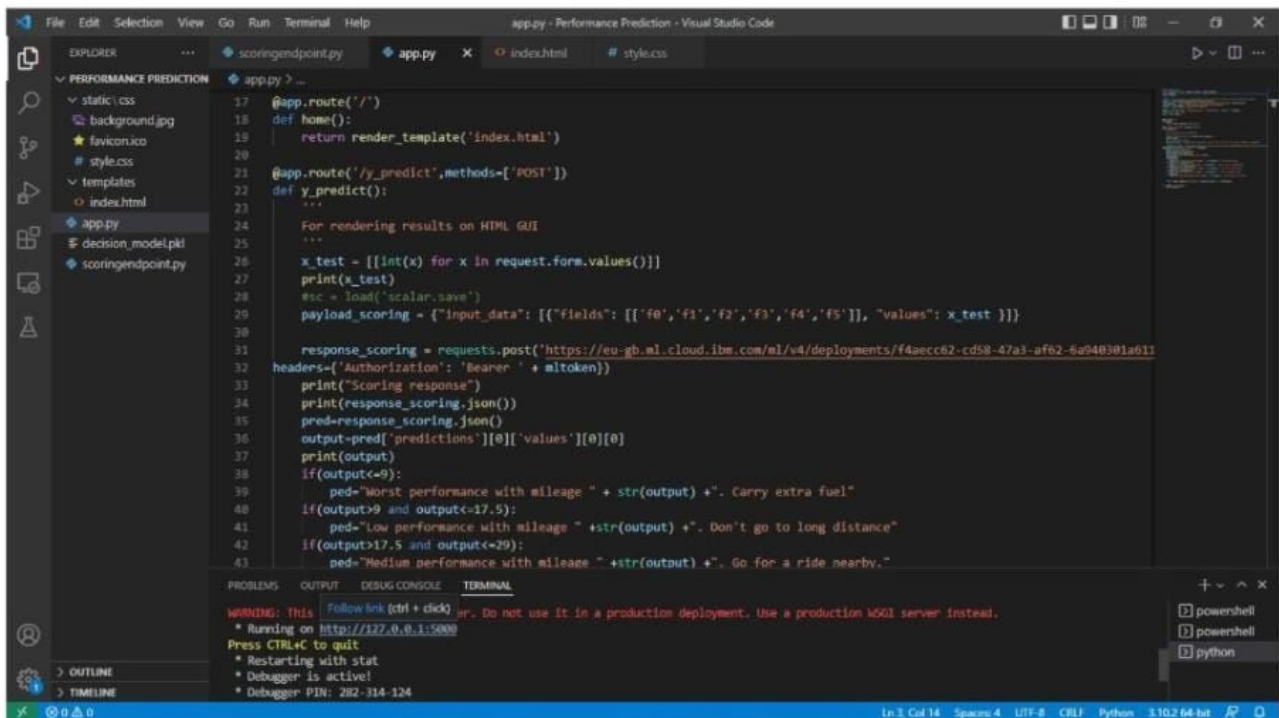
```
scoringendpoint.py > -
1 import requests
2
3 # NOTE: you must manually set API_KEY below using information retrieved from your IBM Cloud account.
4 API_KEY = "YIjAXb1Vp23FvN8FxmHfEECIBjRuptp4aL7jNGzUe"
5 token_response = requests.post("https://iam.cloud.ibm.com/identity/token", data={"apikey":
6 | API_KEY, "grant_type": "urn:ibm:params:oauth:grant-type:apikey"})
7 mltoken = token_response.json()["access_token"]
8
9 header = {"Content-Type": "application/json", "Authorization": "Bearer " + mltoken}
10
11 # NOTE: manually define and pass the array(s) of values to be scored in the next line
12 payload_scoring = {"input_data": [{"fields": [{"f0", "f1", "f2", "f3", "f4", "f5"}], "values": [[0,160,380,3504,82,1]]}]}
13
14 response_scoring = requests.post("https://eu-gb.ml.cloud.ibm.com/ml/v4/deployments/f4a6cc62-cd58-47a3-af62-6a948381a611/pre
15 | headers={"Authorization": "Bearer " + mltoken})
16 print("Scoring response")
17 print(response_scoring.json())
18 pred=response_scoring.json()
19 output=pred['predictions'][0]['values'][0][0]
20 print(output)
```

PS D:\ibm\sprint\ibm cloud\Performance Prediction> python scoringendpoint.py

Scoring response

```
{
  "predictions": [{"fields": [{"prediction": 17.7}]}]}
```

PS D:\ibm\sprint\ibm cloud\Performance Prediction>



```
app.py > -
17 @app.route('/')
18 def home():
19     return render_template("index.html")
20
21 @app.route('/y_predict', methods=['POST'])
22 def y_predict():
23     ...
24     For rendering results on HTML GUI
25     ...
26     x_test = [[int(x) for x in request.form.values()]]
27     print(x_test)
28     #sc = load('scalar.sav')
29     payload_scoring = {"input_data": [{"fields": [{"f0", "f1", "f2", "f3", "f4", "f5"}], "values": x_test }]}
30
31     response_scoring = requests.post("https://eu-gb.ml.cloud.ibm.com/ml/v4/deployments/f4a6cc62-cd58-47a3-af62-6a948381a611
32 | headers={"Authorization": "Bearer " + mltoken})
33 print("Scoring response")
34 print(response_scoring.json())
35 pred=response_scoring.json()
36 output=pred['predictions'][0]['values'][0][0]
37 print(output)
38 if(output<9):
39     ped="Worst performance with mileage " + str(output) + ". Carry extra fuel"
40
41 if(output>9 and output<17.5):
42     ped="Low performance with mileage " + str(output) + ". Don't go to long distance"
43
44 if(output>17.5 and output<29):
45     ped="Medium performance with mileage " + str(output) + ". Go for a ride nearby."
```

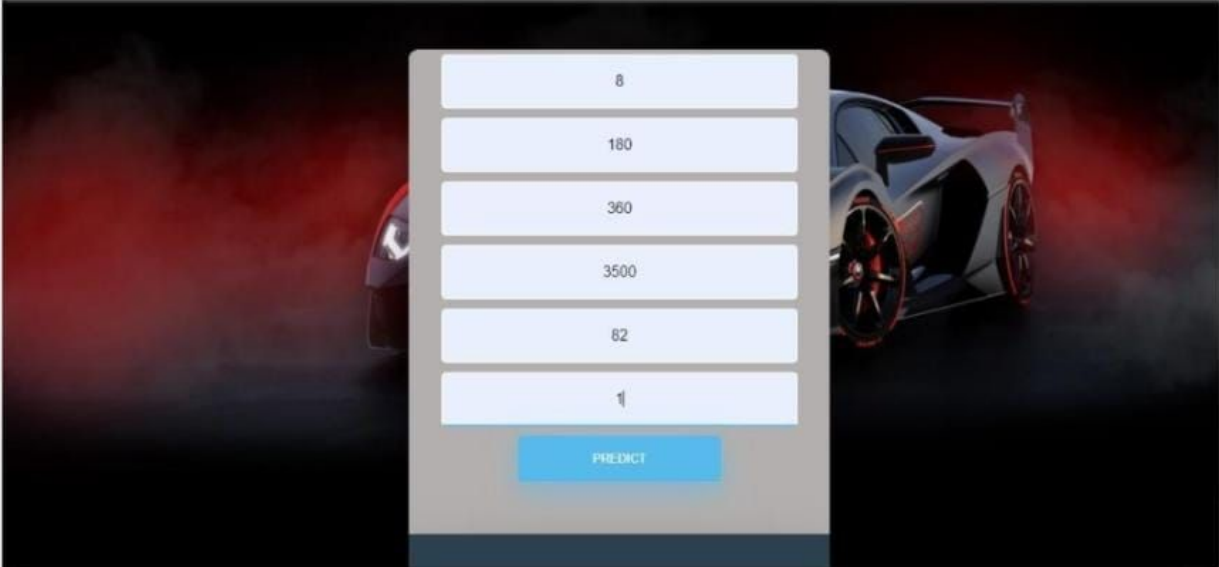
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.

* Running on <http://127.0.0.1:5000>

Press CTRL+C to quit

- * Restarting with stat
- * Debugger is active!
- * Debugger PIN: 282-314-124

PREDICT YOUR CAR'S PERFORMANCE



8

180

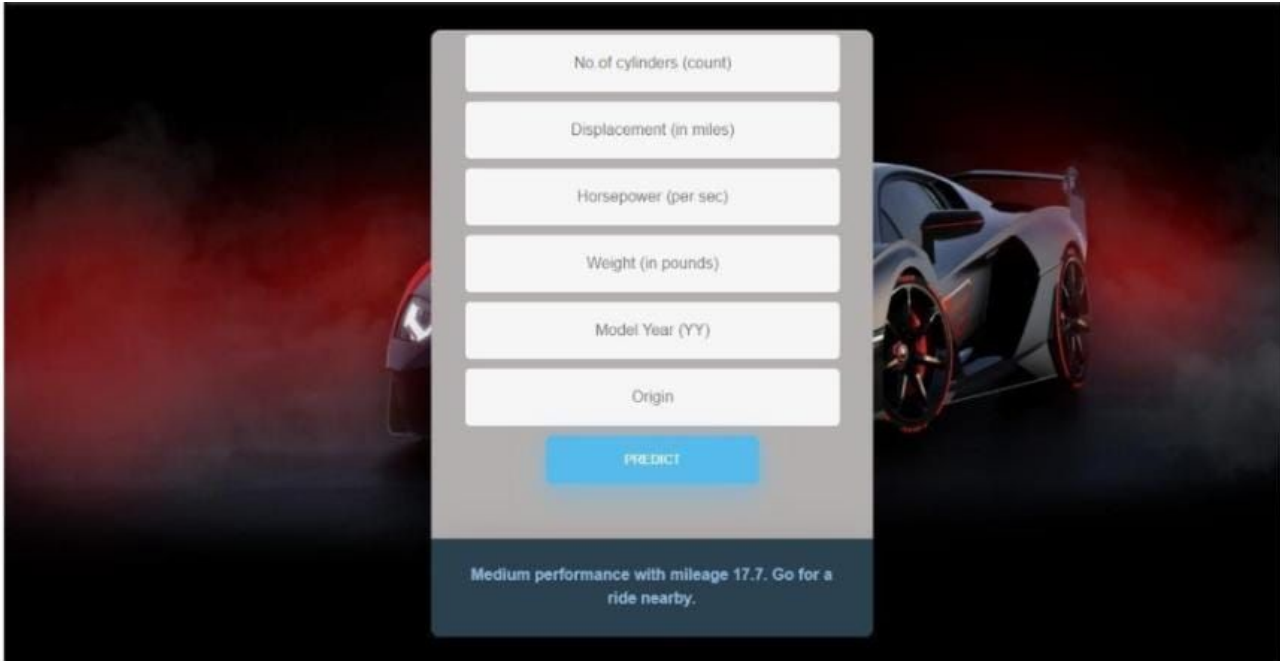
360

3500

82

1

PREDICT



No. of cylinders (count)

Displacement (in miles)

Horsepower (per sec)

Weight (in pounds)

Model Year (YY)

Origin

PREDICT

Medium performance with mileage 17.7. Go for a ride nearby.