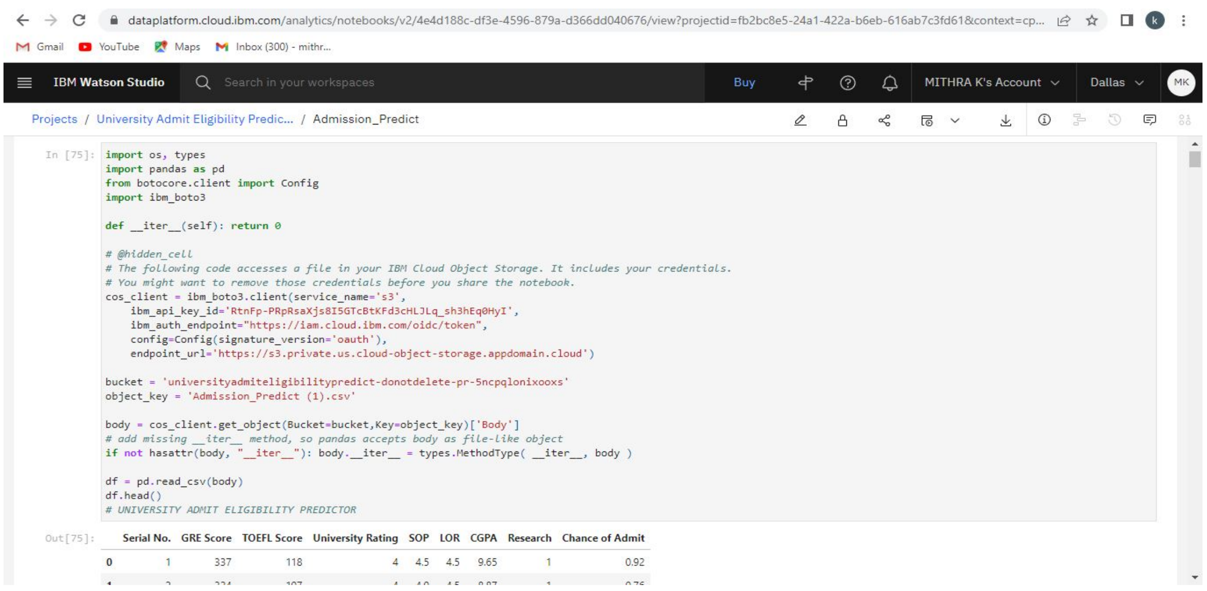
PROJECT DEVELOPMENT PHASE

SPRINT – 4

IBM CLOUD DEPLOYMENT



dataplatform.cloud.ibm.com/analytics/notebooks/v2/4e4d188c-df3e-4596-879a-d366dd040676/view?projectid=fb2bc8e5-24a1-422a-b6eb-616ab7c3fd61&context=cp...

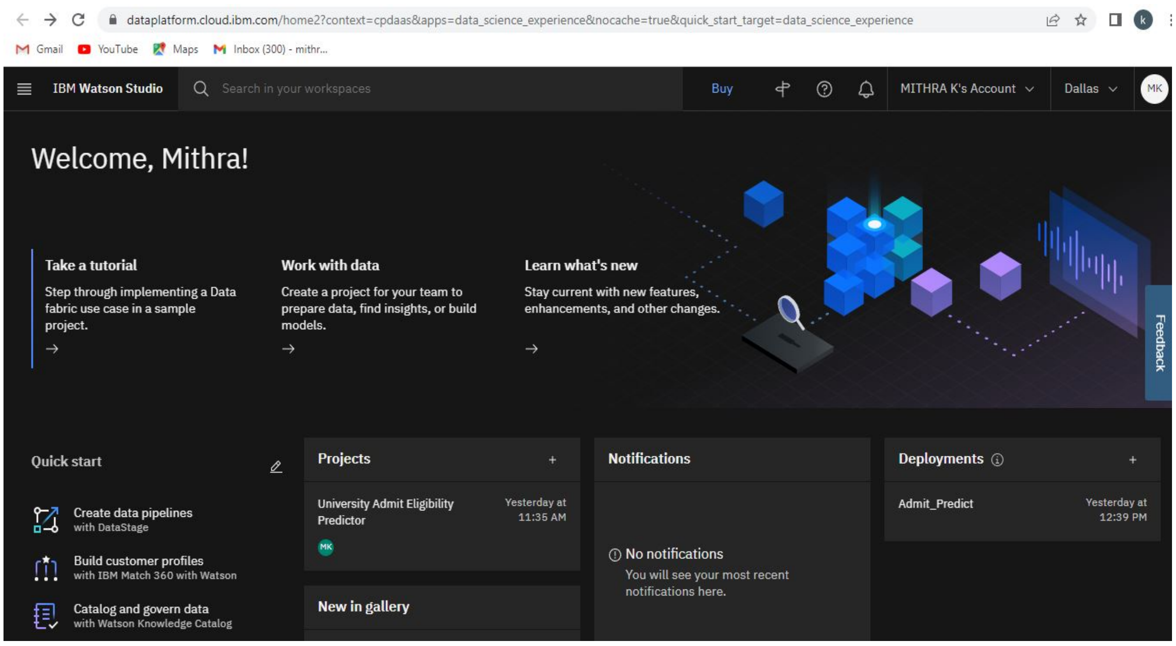
IBM Watson Studio | MITHRA K's Account | Dallas

Projects / University Admit Eligibility Predic... / Admission_Predict

```
In [75]: import os, types
         import pandas as pd
         from botocore.client import Config
         import ibm_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 = ibm_boto3.client(service_name='s3',
             ibm_api_key_id='RtnFp-PRpRsaXjs8I5GTcBtKFd3cHLJLq_sh3hEq0HyI',
             ibm_auth_endpoint="https://iam.cloud.ibm.com/oidc/token",
             config=Config(signature_version='oauth'),
             endpoint_url='https://s3.private.us.cloud-object-storage.appdomain.cloud')

         bucket = 'universityadmiteligibilitypredict-donotdelete-pr-5ncpqlonixooxs'
         object_key = 'Admission_Predict (1).csv'

         body = cos_client.get_object(Bucket=bucket,Key=object_key)['Body']
         # add missing __iter__ method, so pandas accepts body as file-like object
         if not hasattr(body, "__iter__"): body.__iter__ = types.MethodType( __iter__, body )

         df = pd.read_csv(body)
         df.head()
         # UNIVERSITY ADMIT ELIGIBILITY PREDICTOR
```

Out[75]:

	Serial No.	GRE Score	TOEFL Score	University Rating	SOP	LOR	CGPA	Research	Chance of Admit
0	1	337	118	4	4.5	4.5	9.65	1	0.92



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

IBM Watson Studio | MITHRA K's Account | Dallas

Welcome, Mithra!

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.

Learn what's new
Stay current with new features, enhancements, and other changes.

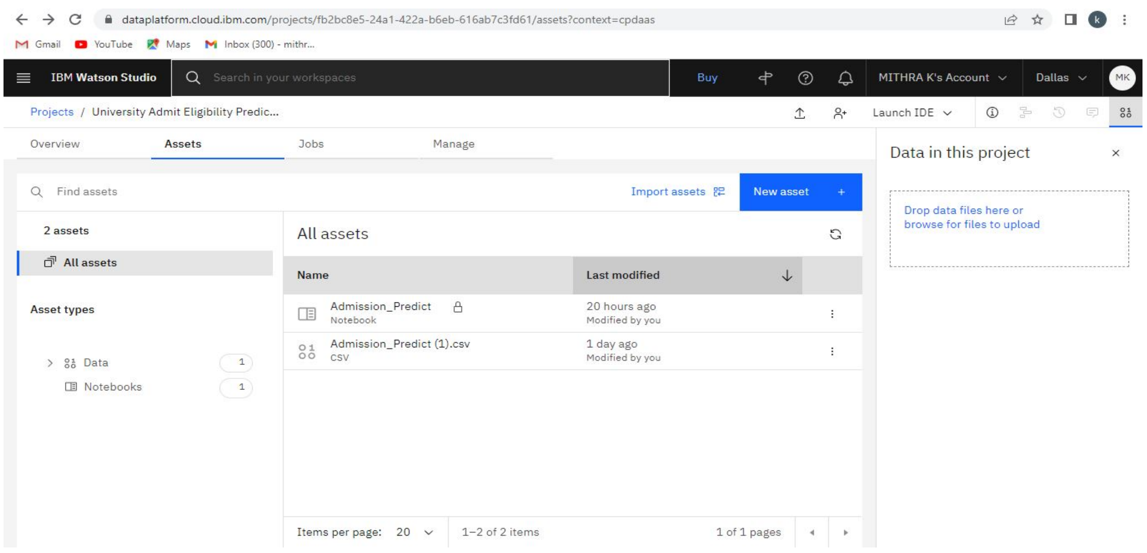
Quick start
Create data pipelines with DataStage
Build customer profiles with IBM Match 360 with Watson
Catalog and govern data with Watson Knowledge Catalog

Projects
University Admit Eligibility Predictor — Yesterday at 11:35 AM

New in gallery

Notifications
No notifications
You will see your most recent notifications here.

Deployments
Admit_Predict — Yesterday at 12:39 PM



Name	Last modified
Admission_Predict Notebook	20 hours ago Modified by you
Admission_Predict (1).csv CSV	1 day ago Modified by you

IBM_APP.PY

```
import joblib

from flask import Flask , request, render_template

from math import ceil

import requests


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

API_KEY = "ACsP3Bo6BnmCyRBl66ImEMRM6P3BbdmHMeuQcxVBCMBw"

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}

app = Flask(__name__)

model = joblib.load("model.pkl")
```

```
@app.route('/')
def index():
return render_template('index.html')

@app.route('/predict',methods = ['GET','POST'])
def admin():
    gre=(eval(request.form["gre"])-290)/(340-290)
    tofl=(eval(request.form["tofl"])-92)/(120-92)
    rating=(eval(request.form["rating"])-1.0)/4.0
    sop=(float(request.form["sop"])-1.0)/4.0
    lor=(float(request.form["lor"])-1.0)/4.0
    cgpa=(float(request.form["cgpa"])-5.5)/(10-5.6)
    research=request.form["research"]
    if (research=="Yes"):
        research=1
    else:
        research=0

    array_of_input_fields = ['gre', 'tofl', 'rating', 'sop', 'lor', 'cgpa', 'research']
    array_of_values_to_be_scored = [gre, tofl, rating, sop, lor, cgpa, research]
    payload_scoring = {"input_data": [{"fields": [array_of_input_fields],
"values": [array_of_values_to_be_scored]}]}
    response_scoring = requests.post('https://us-
south.ml.cloud.ibm.com/ml/v4/deployments/8244ff58-07bb-481d-becc-
c4ad94411dc5/predictions?version=2022-11-18', json=payload_scoring,
headers={'Authorization': 'Bearer ' + mltoken})
    prediction = response_scoring.json()['predictions'][0]['values'][0][0]

    if (prediction > 0.5):
        return render_template("chance.html",p=str(ceil(prediction *100)) +" %")
    return render_template("nochance.html")
if __name__ == '__main__':
    app.run(debug = True, port=5500)
```