

## PHASE 3: Project Development Part-1

### Project title:

Machine learning model deployment with IBM cloud Watson studio

### Problem Statement:

Train machine learning models to predict the outcomes in real time. Deploy the models as web services and integrate them into your applications.

### Project Overview:

House Price Prediction Analysis aims to use Machine learning analysis algorithms to predict the price of houses based on their features like number of rooms, number of bedrooms, age of the house, population of the respective area where the house is located, location of the house and the area income with other relevant factors if available. By this Machine Learning model user can predict the price of the house that can be sold.

### About IBM cloud Watson Studio:

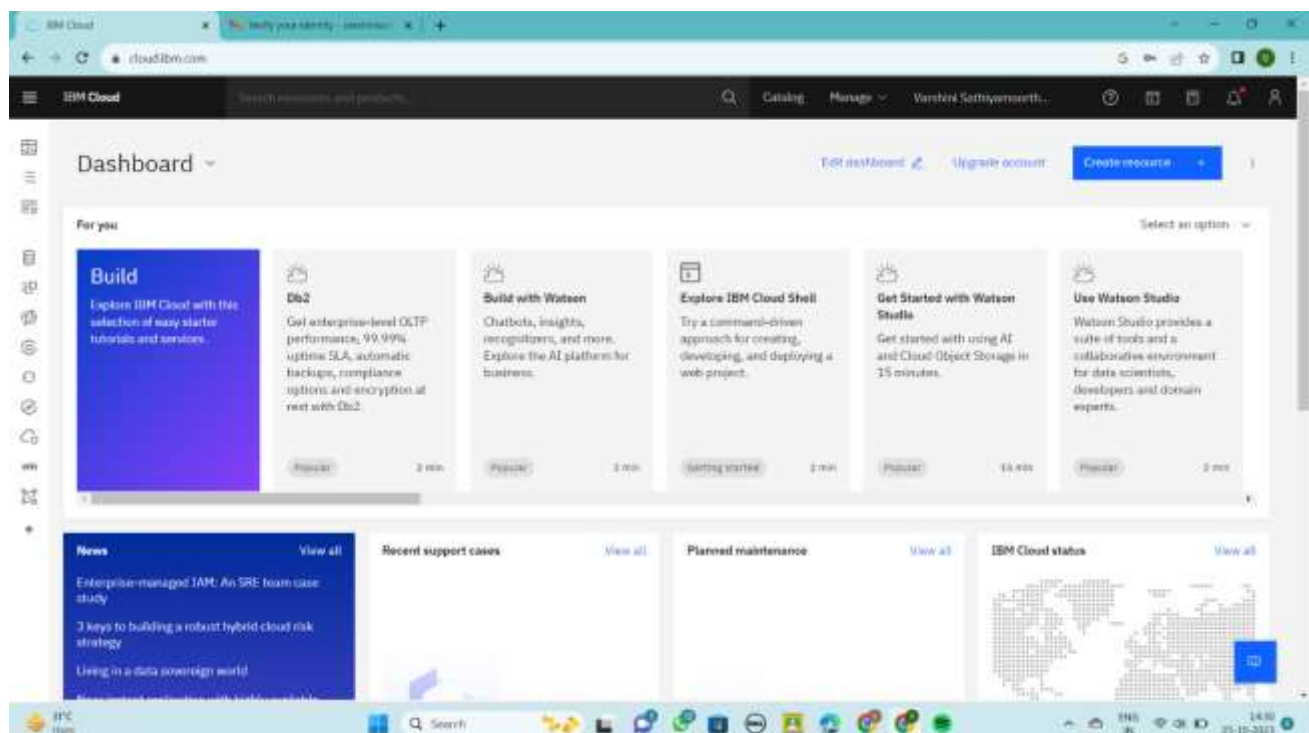
IBM Watson Studio provides tools for data scientists, application developers and subject matter experts to collaboratively and easily work with data to build and train models at scale. It gives us the flexibility to build models where our data

resides and deploy anywhere in a hybrid environment so we can operationalize data science faster.

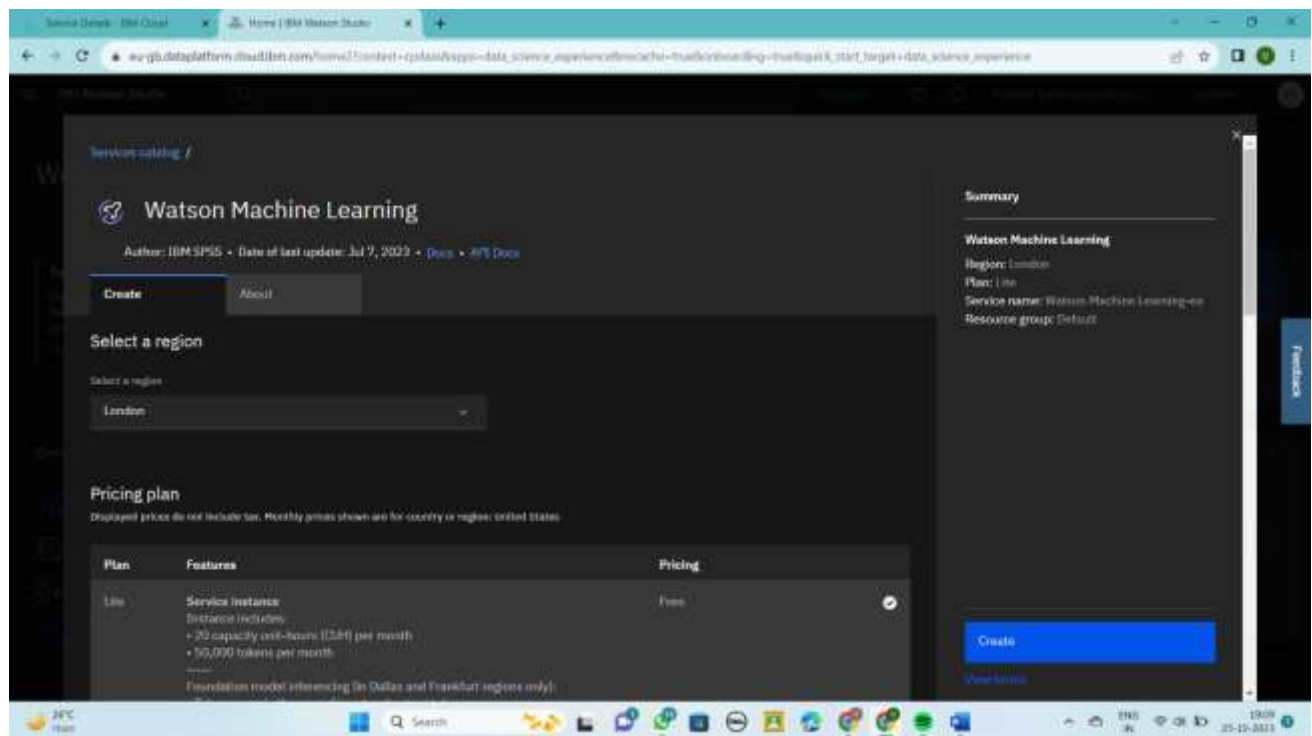
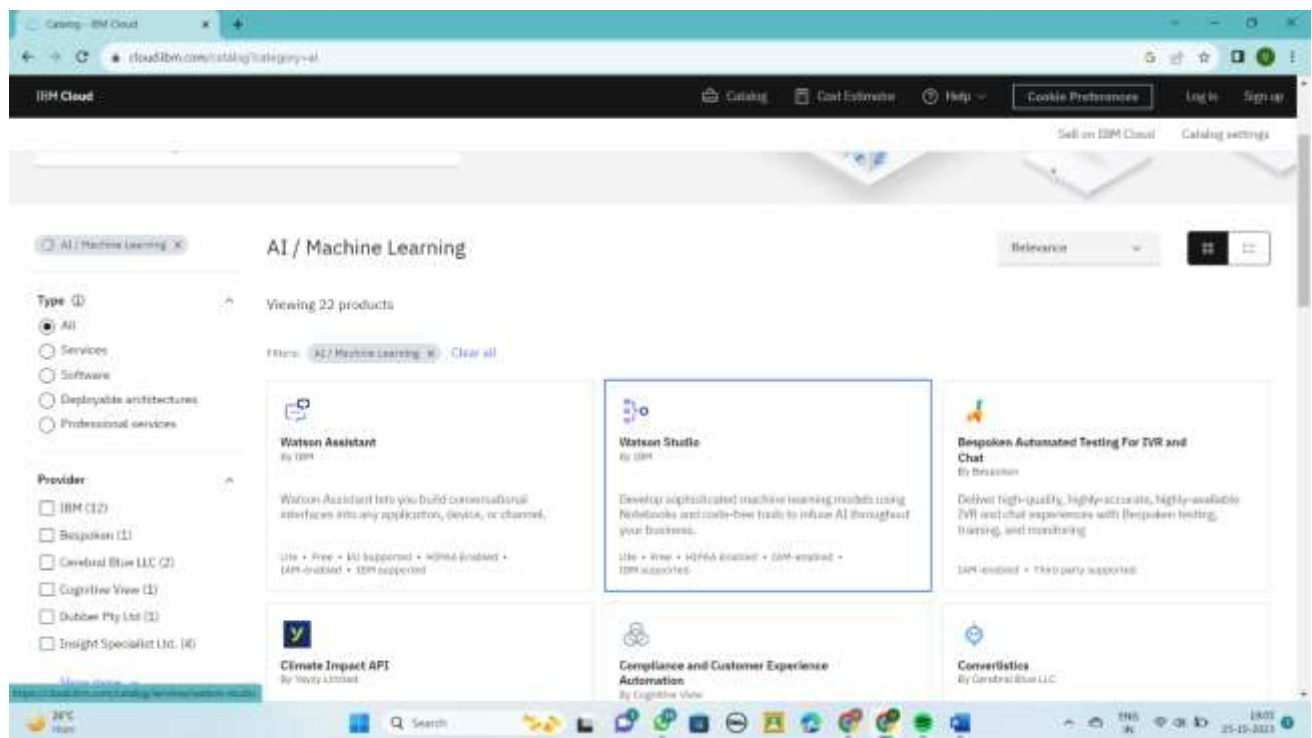
IBM Watson Studio provides various tools for designing, training, and managing machine learning models: **Model builder, Flow editor, Experiment builder, Notebooks, Machine learning command line interface**. Among them, we use notebooks for the working with our dataset.

Stepwise procedure for building the machine learning model using IBM Cloud Watson Studio is as follows:

### Step1: Login to IBM cloud



**Step 2:** Go to catalog and create a Watson Studio service in AI/Machine learning category.



### Step 3: Create a project in IBM Watson Studio Dashboard and assign a Cloud object Storage service to manage datasets

The screenshot shows the 'New project' form in the IBM Watson Studio dashboard. The form is divided into two main sections: 'Define details' and 'Define storage'.

**Define details:**

- Name:** A text input field containing 'House price prediction'.
- Description (optional):** A text area with the placeholder text 'What's the purpose of this project?'.
- Controls:**
  - ☒ Restrict who can be a collaborator (i)
  - ☐ Mark as sensitive (i)

**Define storage:**

- Project includes integration with [Cloud Object Storage](#) for storing project assets.
- 1 Select storage service:** A button labeled 'Add' with the text 'Add an object storage instance, and then return to this page and click Defaults.'
- 2 Defaults:** A button labeled 'Defaults'.

At the bottom right of the form are 'Cancel' and 'Create' buttons.

The screenshot shows the 'Cloud Object Storage' service catalog page in the IBM Watson Studio dashboard. The page is divided into two main sections: 'Summary' and 'Configure your resource'.

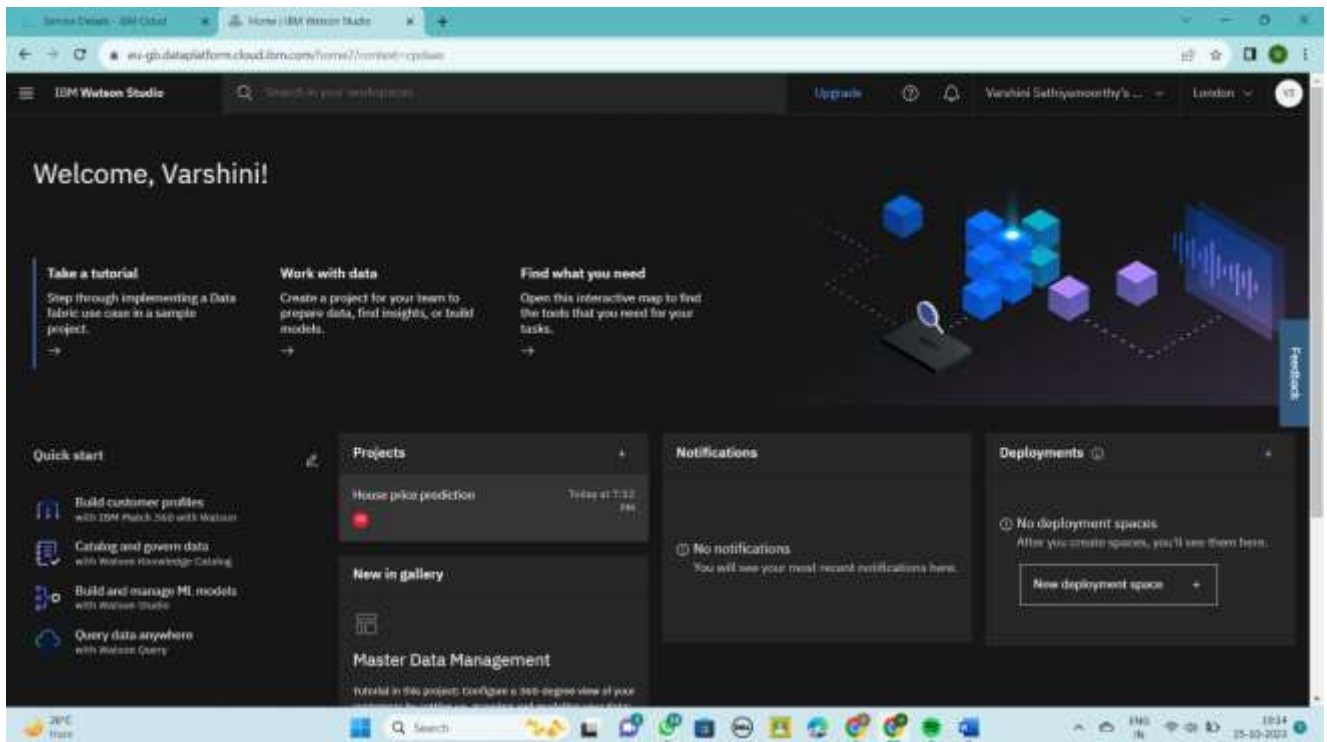
**Summary:**

- Cloud Object Storage**
- Region: Global
- Plan: 1 (0)
- Service name: Cloud Object Storage-ly
- Resource group: Default

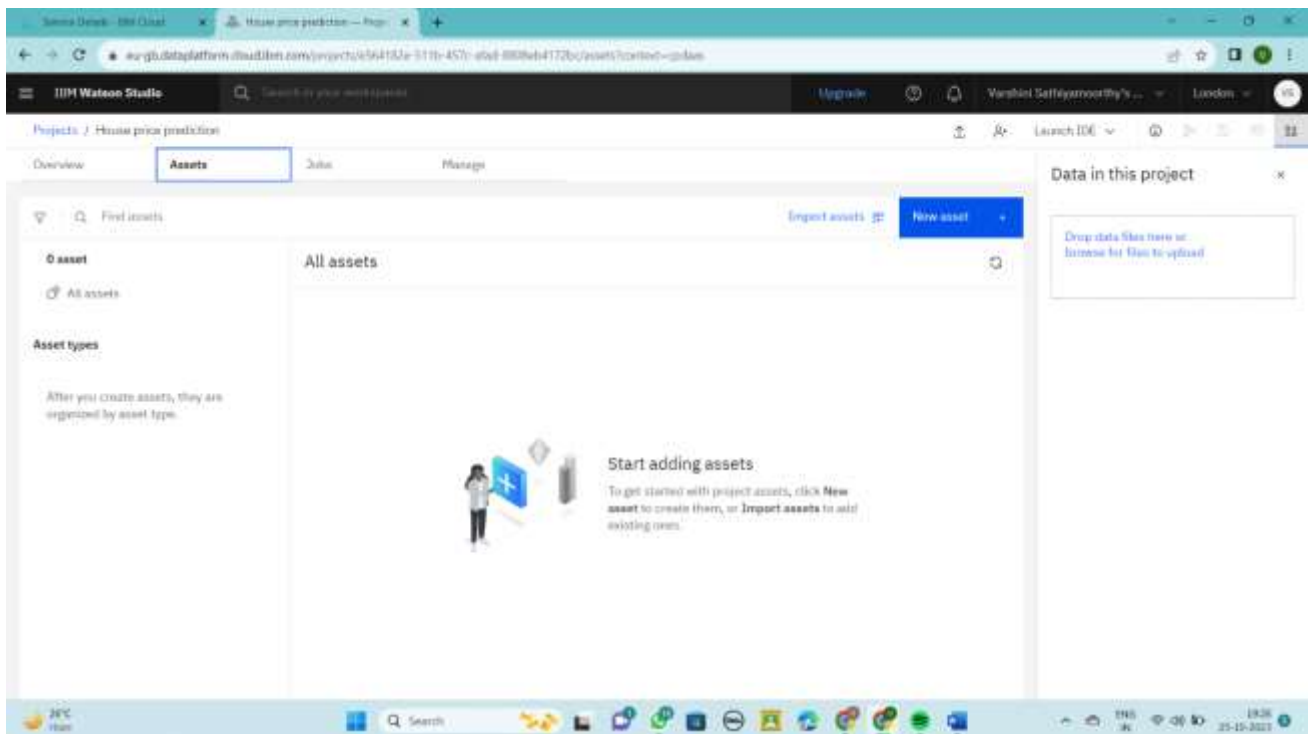
**Configure your resource:**

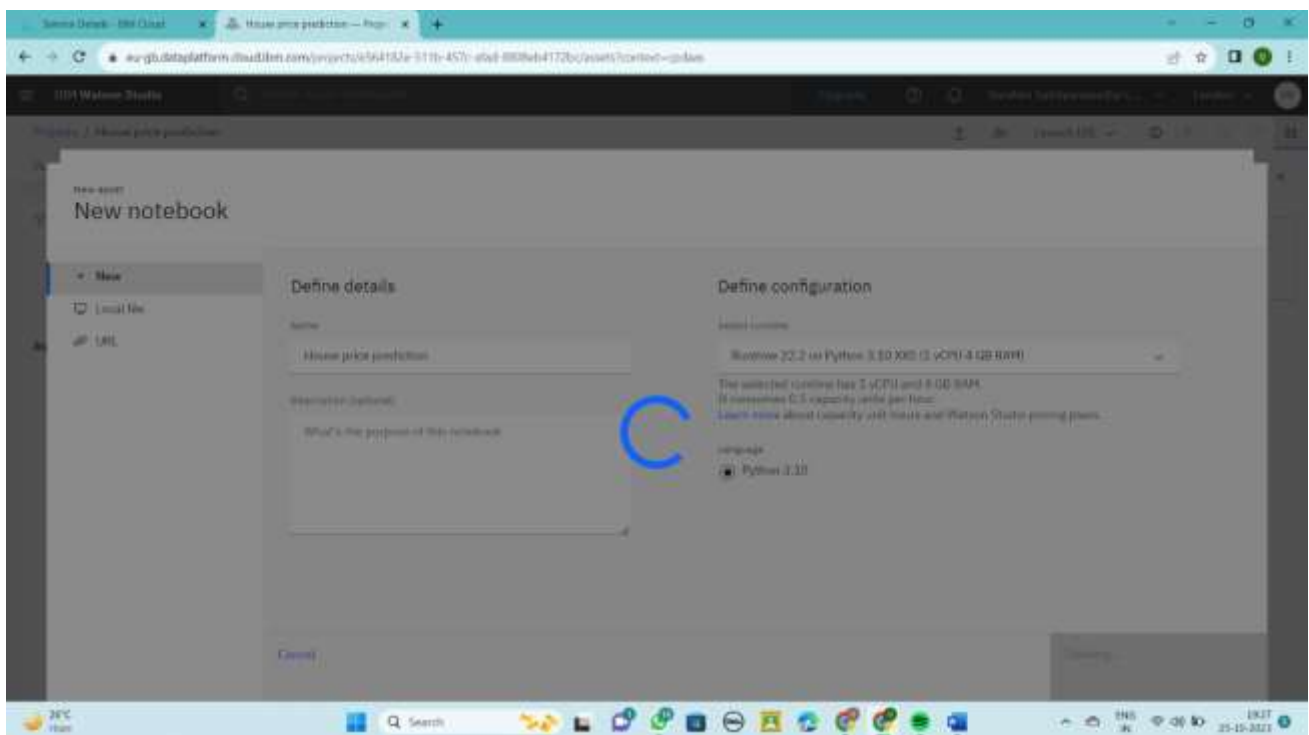
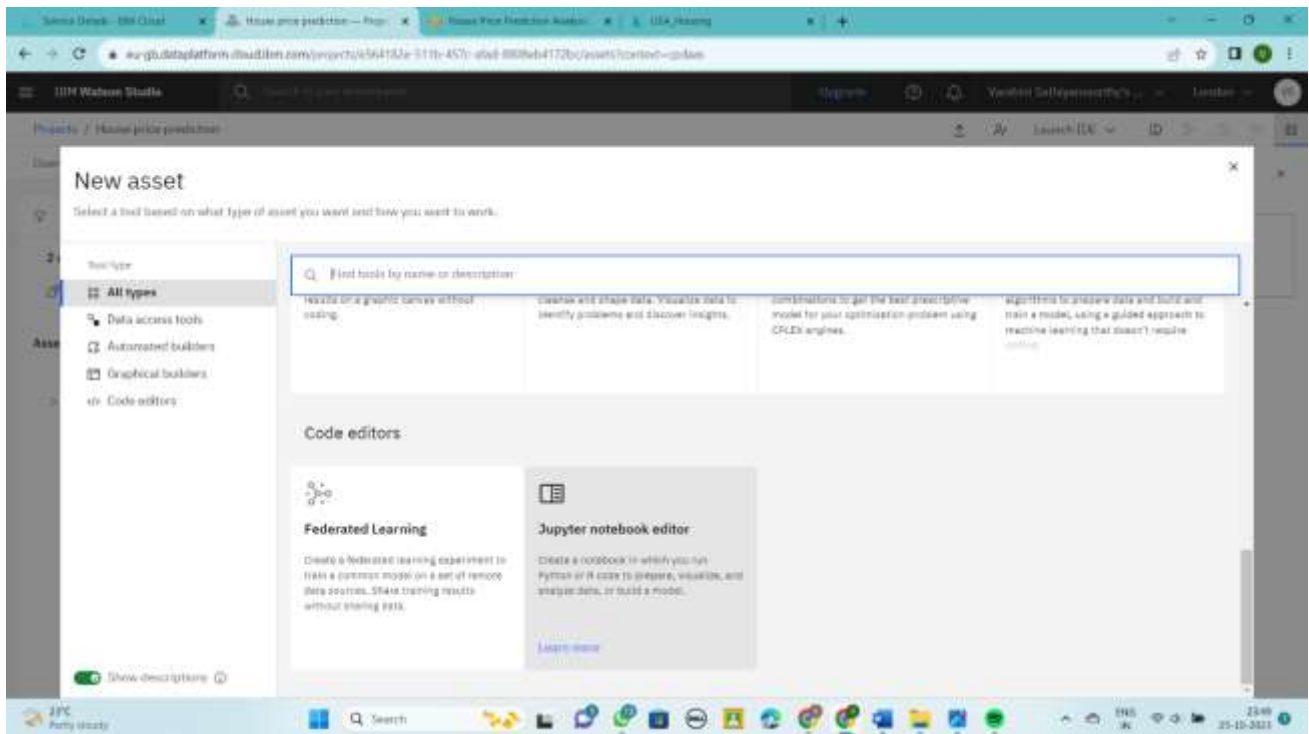
- Service label:** A dropdown menu with 'Cloud Object Storage-ly' selected.
- Select a resource group:** A dropdown menu with 'Default' selected.
- Tags:** A text input field with the placeholder text 'Examples: env/dev, version-1'.

At the bottom right of the page is a 'View items' button.



**Step 4:** Add a jupyter notebook instance in your project to Develop and Deploy Machine Learning Model.

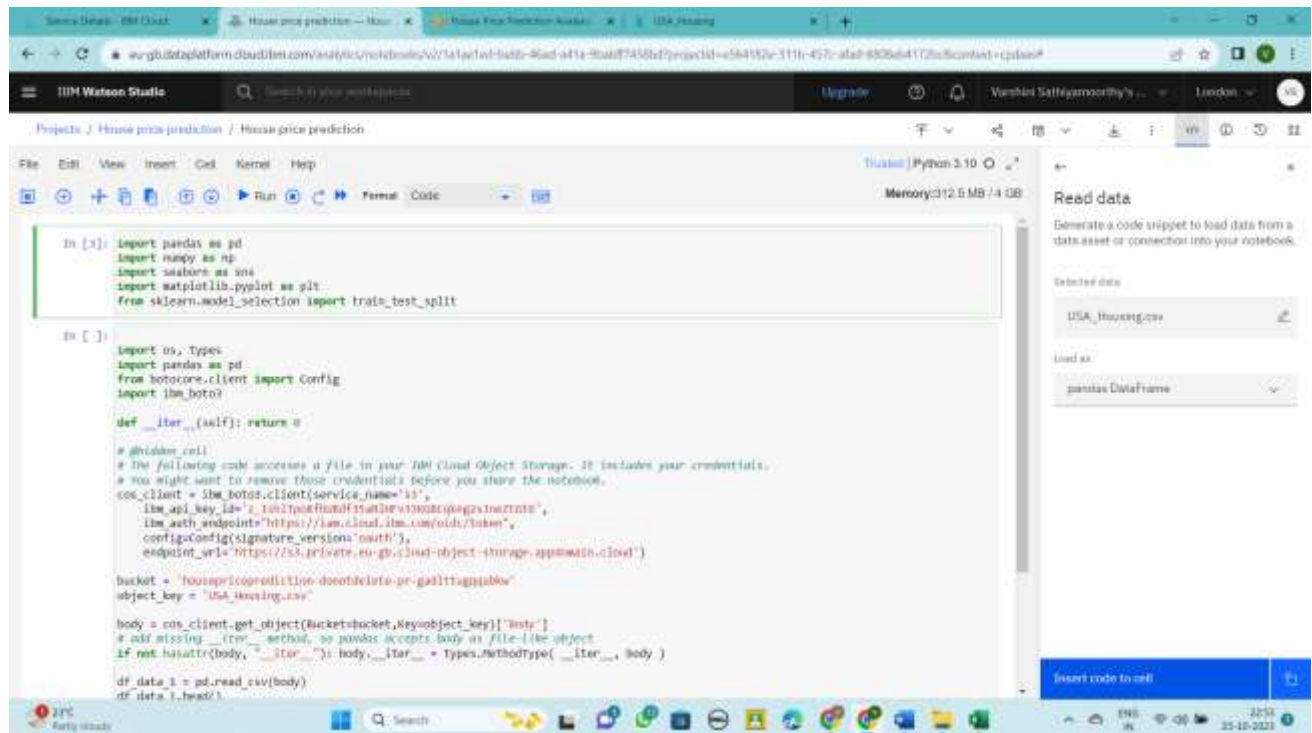




**Step 5:** Build a machine learning model using jupyter notebook instance.



Import the required libraries. In the next step we are going to upload and insert the dataset for training our Machine Learning model as pandas dataframe.



The screenshot shows the IBM Watson Studio interface. The notebook is titled "House price prediction". The first code cell contains the following Python code:

```
In [ ]: import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split

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

def __iter__(self): return 0

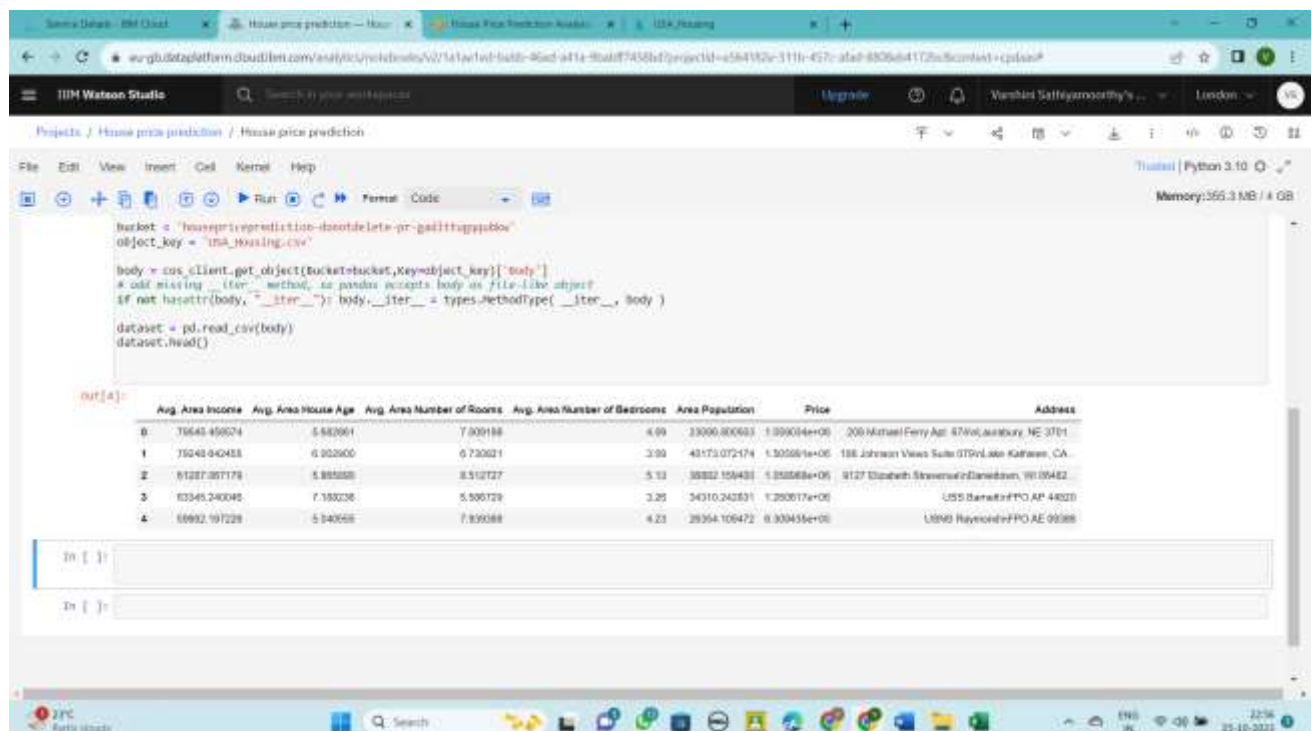
# glider cell
# The following code accesses a file in your IBM Cloud Object Storage. It includes your credentials,
# you might want to remove those credentials before you share the notebook.
cos_client = boto3.client(service_name='s3',
    aws_api_key_id='i_1u62tpatfuthf15an3ev1300tr0qag2s1u2rnt0',
    aws_auth_endpoint='https://iam.cloud.ibm.com/oidc/token',
    config=Config(signature_version='auth'),
    endpoint_url='https://s3.private.eu-gb.cloud-object-storage.appdomain.cloud')

bucket = 'housepriceprediction-dooefdelefa-pr-gallttagppubw'
object_key = 'USA_Housing.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_data = pd.read_csv(body)
df_data.head()
```

The right sidebar shows the "Read data" section with "USA\_Housing.csv" selected and "Load as" set to "pandas DataFrame". A button "Insert code to cell" is visible.



The screenshot shows the second code cell in the notebook, which contains the following Python code:

```
bucket = 'housepriceprediction-dooefdelefa-pr-gallttagppubw'
object_key = 'USA_Housing.csv'

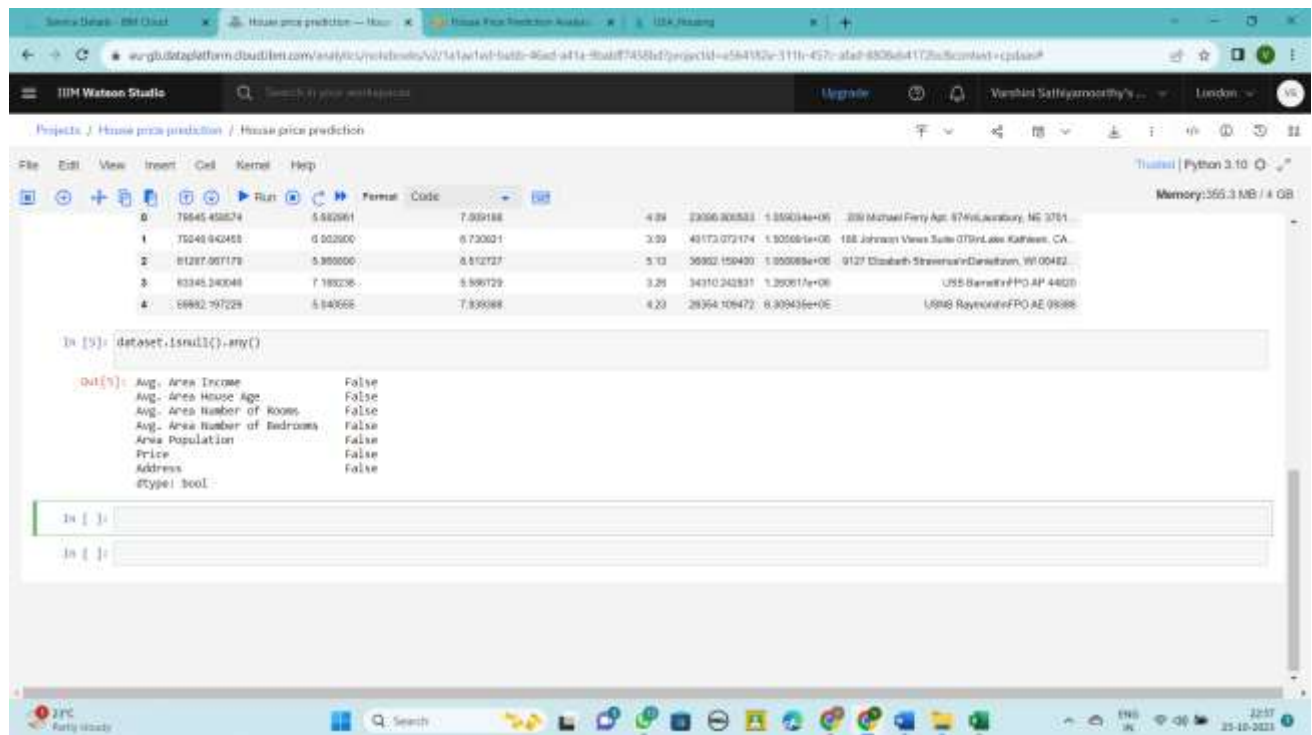
body = cos_client.get_object(bucket=bucket, Key=object_key)['Body']
# add missing __iter__ method, so pandas accepts body as file-like object
if not hasattr(body, '__iter__'): body.__iter__ = types.MethodType(__iter__, body)

dataset = pd.read_csv(body)
dataset.head()
```

The output of the code cell is displayed as a table with 5 columns: Avg. Area Income, Avg. Area House Age, Avg. Area Number of Rooms, Avg. Area Number of Bedrooms, Area Population, Price, and Address. The first 5 rows of data are shown:

	Avg. Area Income	Avg. Area House Age	Avg. Area Number of Rooms	Avg. Area Number of Bedrooms	Area Population	Price	Address
0	70643.45024	5.582061	7.009188	4.99	23000.800603	1.990034e+06	200 Mutual Ferry Apt 87Wic, westbury, NE 3701
1	75040.94048	6.902900	6.730021	3.99	43173.072174	1.300891e+06	188 Johnson Views Suite 07P, Los Angeles, CA
2	81287.867179	5.885888	8.510727	5.13	38882.159430	1.350588e+06	9127 Elizabeth Stoverside, Dannebrog, WI 09482
3	80345.240046	7.183238	5.506729	3.25	34310.242831	1.250017e+06	US55 Barakinf P, AP 44020
4	68682.997228	5.340568	7.839588	4.23	28354.106472	8.930435e+05	US80 Raymond P, PO AE 09388

Once the dataset is imported we can proceed further with pre-processing steps and building the model as follows.

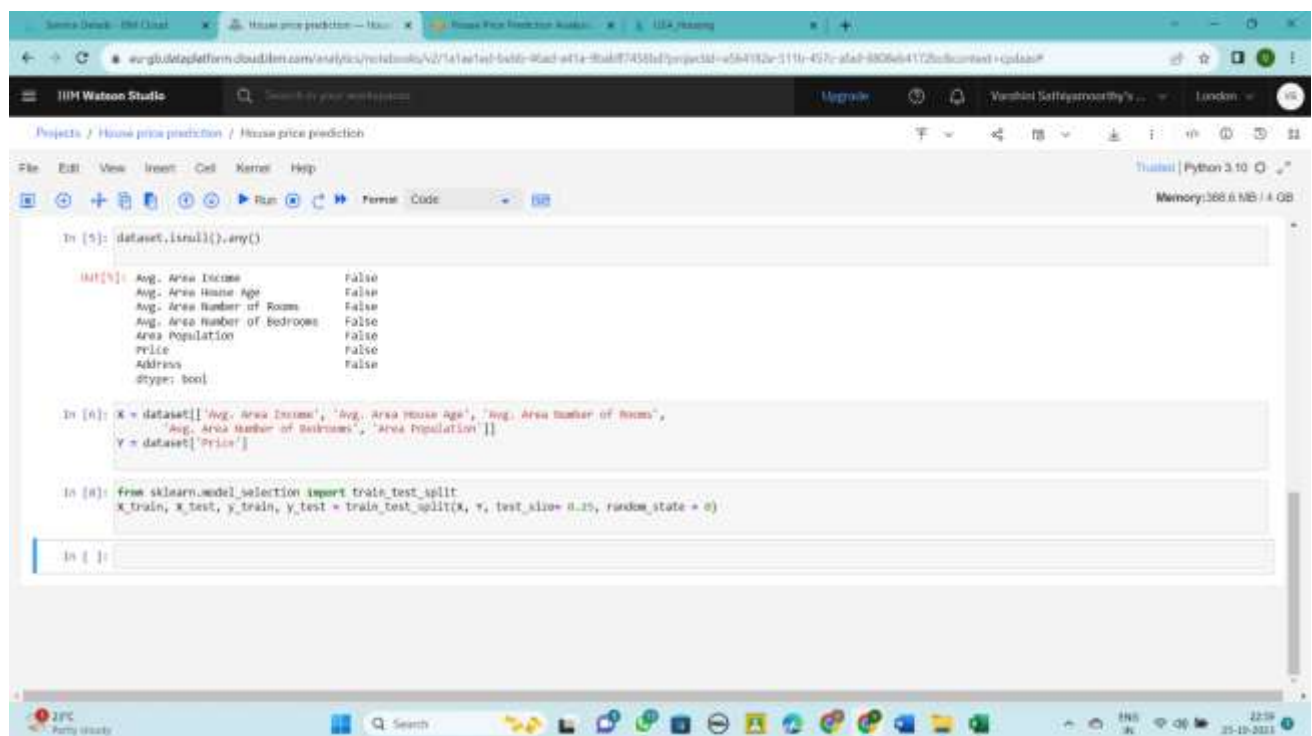


The screenshot shows the IBM Watson Studio interface. At the top, there's a browser window with the URL <https://cloud.ibm.com/analytics/uc4b2b0a527141a1e1ed-b4b6-96ad-4414-8ba0f74581d7/projectId=e5b4192e-311b-457c-adaf-8305b4112bdc/context=cpldaaP>. Below the browser, the IBM Watson Studio logo is visible. The main workspace shows a project named "House price prediction" with a sub-project "House price prediction". The "Data" tab is selected, displaying a table with 5 rows and 10 columns. The columns are: "id", "lat", "lon", "area", "price", "rooms", "age", "population", "address", and "type". The first row of data is: 0, 75645.458274, 5.582061, 7.932188, 4.24, 23050.30533, 1.55034e+05, 209 Michael Perry Apt. 8746, Leicestershire, LE 3761, 188 Johnson Vines Suite 0709, Los Angeles, CA, 9127 Elizabeth Stevens, W 00482, USS Barnett P.O. AP 4402, US96 Raymonte P.O. AE 9888.

```
In [5]: dataset.isnull().any()
```

```
Out[5]: Avg. Area Income      False
Avg. Area House Age      False
Avg. Area Number of Rooms  False
Avg. Area Number of Bedrooms False
Area Population          False
Price                   False
Address                 False
dtype: bool
```

Splitting into training and testing datasets



The screenshot shows the IBM Watson Studio interface. At the top, there's a browser window with the URL <https://cloud.ibm.com/analytics/uc4b2b0a527141a1e1ed-b4b6-96ad-4414-8ba0f74581d7/projectId=e5b4192e-311b-457c-adaf-8305b4112bdc/context=cpldaaP>. Below the browser, the IBM Watson Studio logo is visible. The main workspace shows a project named "House price prediction" with a sub-project "House price prediction". The "Data" tab is selected, displaying a table with 5 rows and 10 columns. The columns are: "id", "lat", "lon", "area", "price", "rooms", "age", "population", "address", and "type". The first row of data is: 0, 75645.458274, 5.582061, 7.932188, 4.24, 23050.30533, 1.55034e+05, 209 Michael Perry Apt. 8746, Leicestershire, LE 3761, 188 Johnson Vines Suite 0709, Los Angeles, CA, 9127 Elizabeth Stevens, W 00482, USS Barnett P.O. AP 4402, US96 Raymonte P.O. AE 9888.

```
In [5]: dataset.isnull().any()
```

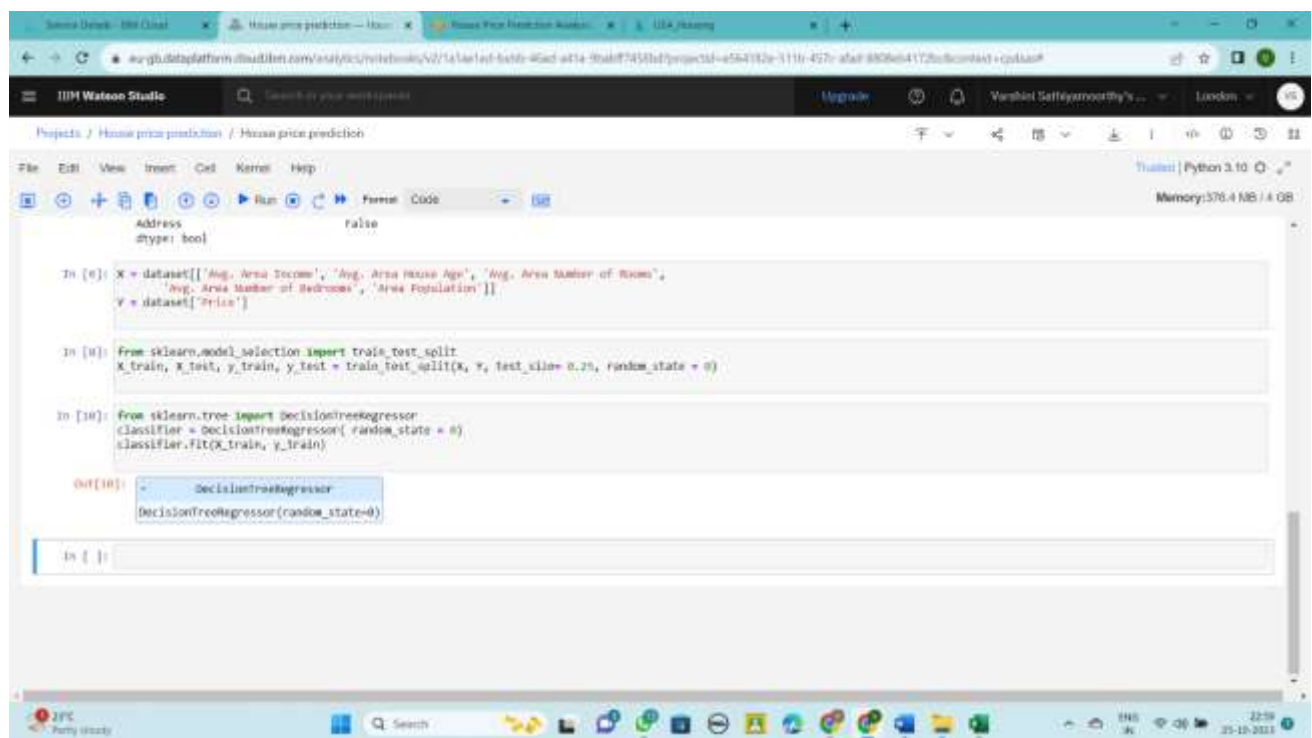
```
Out[5]: Avg. Area Income      False
Avg. Area House Age      False
Avg. Area Number of Rooms  False
Avg. Area Number of Bedrooms False
Area Population          False
Price                   False
Address                 False
dtype: bool
```

```
In [6]: X = dataset[['Avg. Area Income', 'Avg. Area House Age', 'Avg. Area Number of Rooms',
                    'Avg. Area Number of Bedrooms', 'Area Population']]
        Y = dataset['Price']
```

```
In [8]: from sklearn.model_selection import train_test_split
        X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=0.25, random_state=0)
```



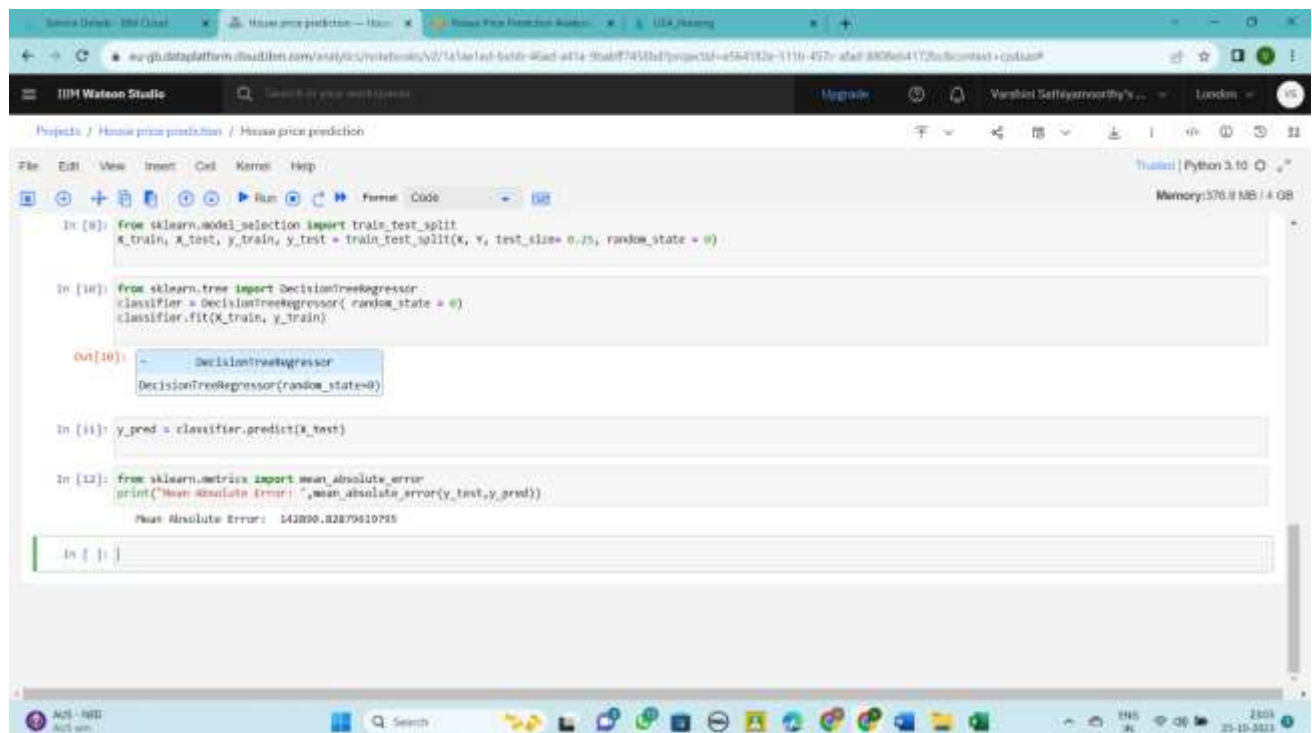
## Fitting Decision Tree Regression to the Training set:



The screenshot shows the IBM Watson Studio interface with a Jupyter Notebook open. The notebook contains the following code:

```
In [6]: X = dataset[['Avg. Area Income', 'Avg. Area House Age', 'Avg. Area Number of Rooms',  
                  'Avg. Area Number of Bedrooms', 'Area Population']]  
y = dataset['Price']  
  
In [8]: from sklearn.model_selection import train_test_split  
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25, random_state=0)  
  
In [10]: from sklearn.tree import DecisionTreeRegressor  
classifier = DecisionTreeRegressor(random_state=0)  
classifier.fit(X_train, y_train)  
  
Out[10]: DecisionTreeRegressor  
DecisionTreeRegressor(random_state=0)
```

## Predicting the Test set results and finding the mean absolute error



The screenshot shows the IBM Watson Studio interface with the Jupyter Notebook updated with the following code:

```
In [8]: from sklearn.model_selection import train_test_split  
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25, random_state=0)  
  
In [10]: from sklearn.tree import DecisionTreeRegressor  
classifier = DecisionTreeRegressor(random_state=0)  
classifier.fit(X_train, y_train)  
  
Out[10]: DecisionTreeRegressor  
DecisionTreeRegressor(random_state=0)  
  
In [11]: y_pred = classifier.predict(X_test)  
  
In [12]: from sklearn.metrics import mean_absolute_error  
print("Mean Absolute Error: ", mean_absolute_error(y_test, y_pred))  
  
Mean Absolute Error: 141890.02879610755
```

Thus the machine learning model was built, trained and tested using IBM Watson Studio.