A
Report on
"Predictive Modeling Analytics"
(CS-18.338)

In partial fulfillment of requirements for the degree of

Bachelor of Technology (B.Tech.)
in
Computer Science and Engineering

Topic: Mobile Price Classification

Submitted by:
Ms. Akshita Kanther(180161)
Ms. Hiral Jain(180167)

Under the Guidance of
Ms. Khushbu Datwani

Department of Computer Science and Engineering
SCHOOL OF ENGINEERING AND TECHNOLOGY
Mody University of Science and Technology
Lakshmangarh, Distt. Sikar-332311
May 2021

ACKNOWLEGMENT

We would like to express our gratitude to our project mentor **Ms. Khushbu Datwani** for her guidance and encouragement for successful completion of this project. Her kindness and help have been the source of encouragement for us. We would also like to extend special thanks to IBM for providing us such a passionate and motivated trainer.

Akshita Kanther

Hiral Jain

TABLE OF CONTENTS

INTRODUCTION

WHAT IS MACHINE LEARNING?

Machine learning is the science of getting computers to act without being explicitly programmed. As it is evident from the name, it gives the computer that makes it more similar to humans: The ability to learn. It focuses on the development of computer programs that can access data and use it to learn for themselves.

APPROACHES TO MACHINE LEARNING:-

- **Supervised learning -** can apply what has been learned in the past to new data using labelled examples to predict future events. Starting from the analysis of a known training dataset, the learning algorithm produces an inferred function to make predictions about the output values. The system is able to provide targets for any new input after sufficient training. The learning algorithm can also compare its output with the correct, intended output and find errors in order to modify the model accordingly.
- **Unsupervised learning -** are used when the information used to train is neither classified nor labelled. Unsupervised learning studies how systems can infer a function to describe a hidden structure from unlabelled data. The system doesn't figure out the right output, but it explores the data and can draw inferences from datasets to describe hidden structures from unlabelled data.
- **Reinforcement learning -** is a learning method that interacts with its environment by producing actions and discovers errors or rewards. Trial and error search and delayed reward are the most relevant characteristics of reinforcement learning. This method allows machines and software agents to automatically determine the ideal behaviour within a specific context in order to maximize its performance. Simple reward feedback is required for the agent to learn which action is best; this is known as the reinforcement signal.

Dataset: Mobile Price Classification

This project is to analyse a dataset related to mobiles and classify the mobile on the basis of different price ranges (low, medium, high and very high). It contains information about battery power, RAM, Bluetooth, Wi-Fi, height, width, price range, etc. of the mobiles of different types and companies. This data set contains 2000 rows and 21 columns.

Features of this project are as follows:-

I. Explore the dataset, Data refining , cleaning and visualizations
II. Visual Recognition Service
III. Implementing the Machine Learning Model on selected data sets and deploying your model with the creation of a Job and testing
IV. Concept of python using Jupyter notebook in IBM Watson

TOOLS AND WORKING ENVIRONMENT-

Project will be made using IBM Watson and Python Notebook on IBM cloud platform.

Python libraries used are as follows:

- Numpy
- Matplotlib
- Seaborn
- Pandas
- Scikit-Learn

DATA PREPROCESSING-

We use Data Refinery to cleanse and shape tabular data with a graphical flow editor. We can also use interactive templates to code operations, functions, and logical operators. When we cleanse data, we fix or remove data that is incorrect, incomplete, improperly formatted, or duplicated. When we shape data, we customize it by filtering, sorting, combining or removing columns, and performing operations.

We create a Data Refinery flow as a set of ordered operations on data. Data Refinery includes a graphical interface to profile our data to validate it and over 20 customizable charts that

give us perspective and insights into our data. When we save the refined data set, we typically load it to a different location than where we read it from. In this way, our source data remains untouched by the refinement process.

The Data Refinery service reduces the amount of time it takes to prepare data. Use pre-defined operations that we can use in our data flows to transform large amounts of raw data into consumable, quality data that's ready for analysis.

- **Required service**

  Watson Studio or Watson Knowledge Catalog
- **Data format**

  Avro, CSV, JSON, Parquet, TSV (read only), or delimited text data asset

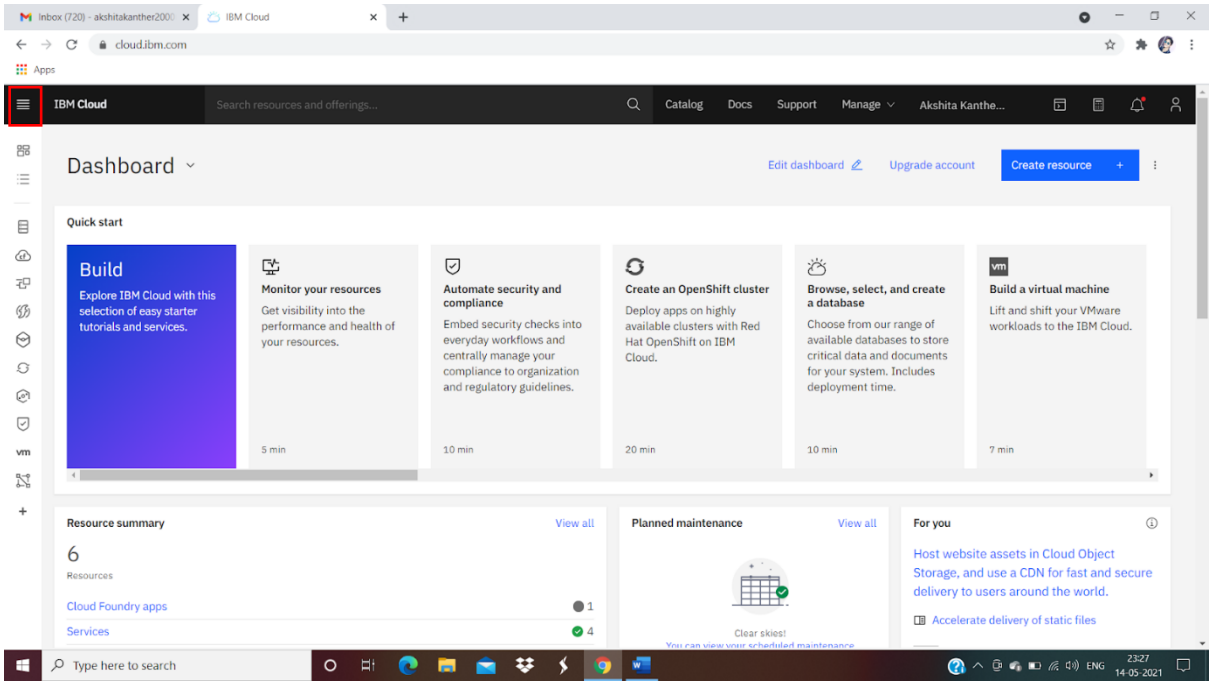
  Tables in relational data sources
- **Data size**

  Any. Data Refinery operates on a sample subset of rows in the data set. The sample size is 1 MB or 10,000 rows, whichever comes first. However, when you run a job for the Data Refinery flow, the entire data set is processed.

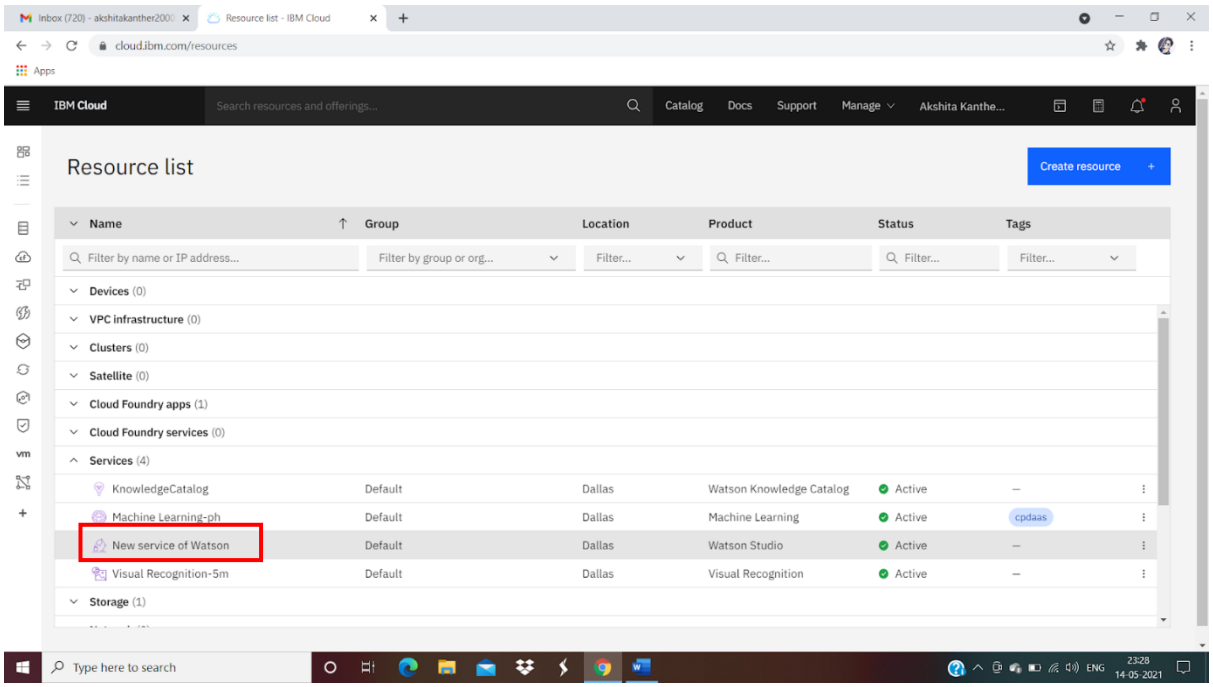
With Data Refinery, we can:

- Interactively discover, cleanse, and transform our data with over 100 built-in operations. No coding is required.
- Understand the quality and distribution of our data using dozens of built-in charts, graphs, and statistics.
- Automatically detect data types and business classifications.
- Schedule data flow executions for repeatable outcomes.

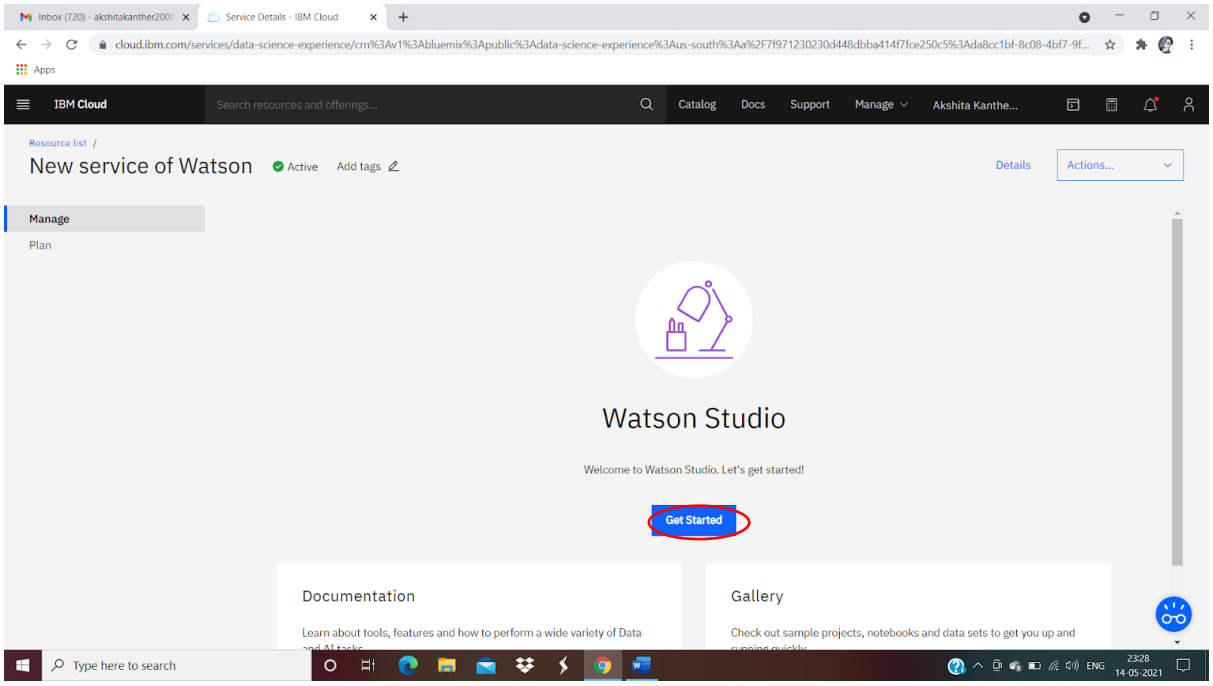
Step-1. Login to the IBM cloud platform and click on Services under Resource summary or click on Resource list from the Navigation pane on the left side of the screen.



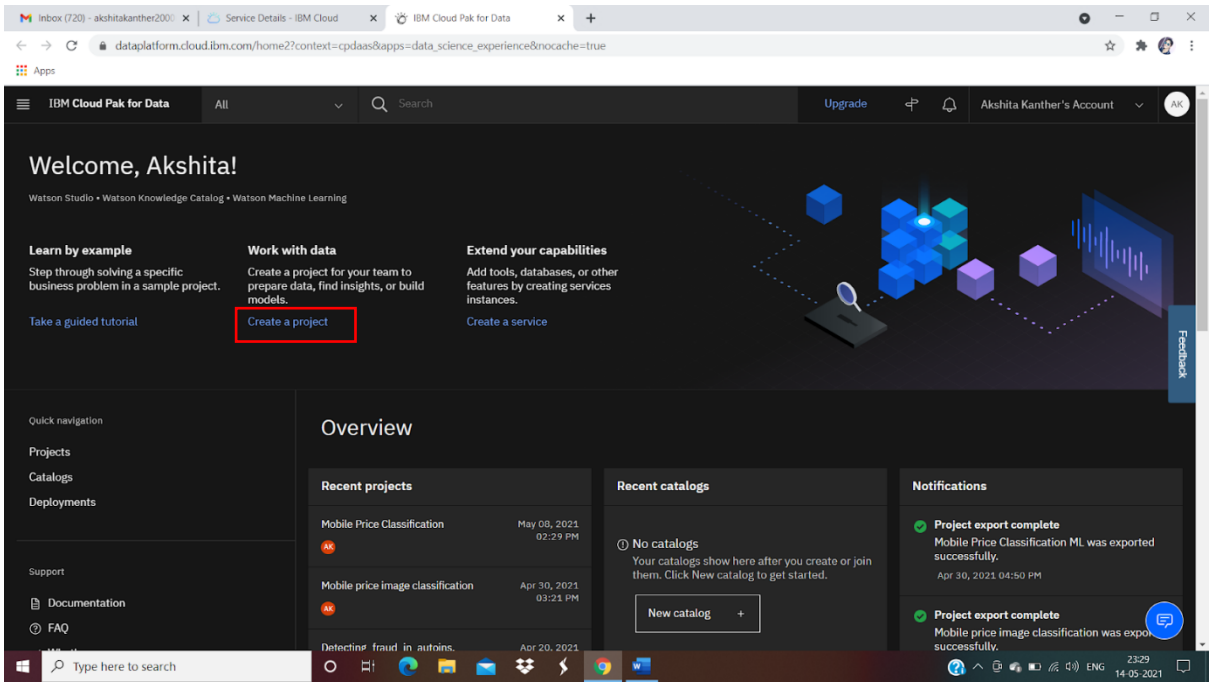
Step-2. Click on Watson service instance that is previously created. If Watson service instance is not previously created then create it.



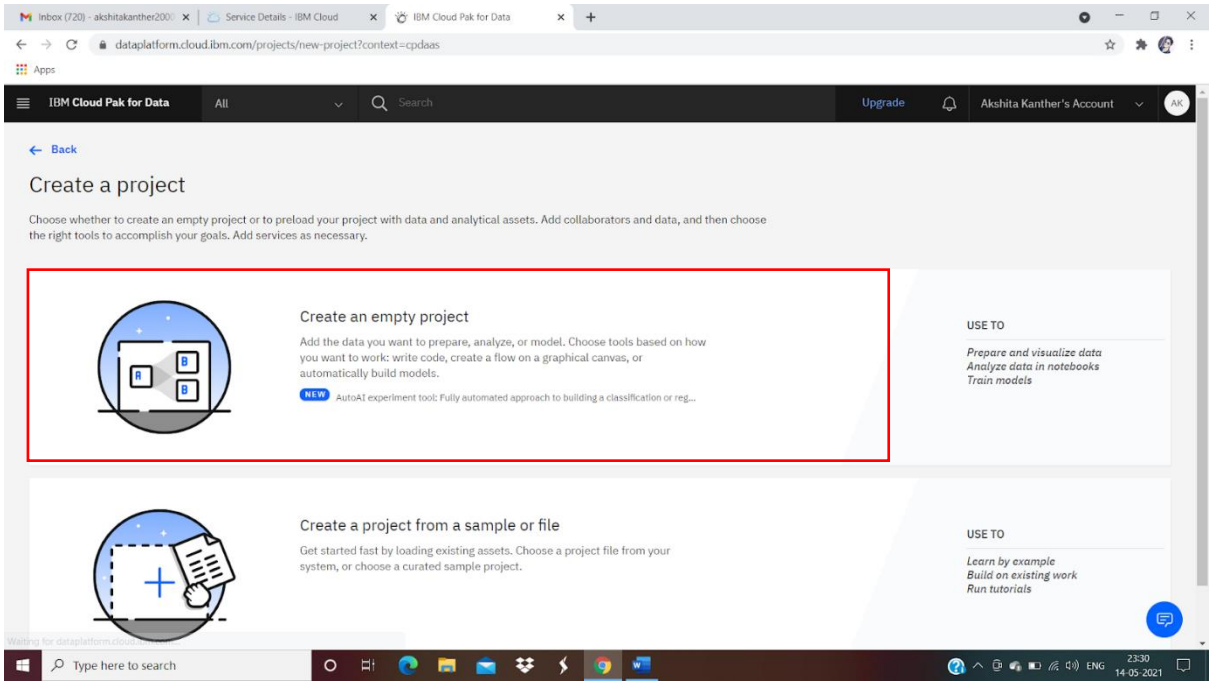
Step-3. Click on Get Started.



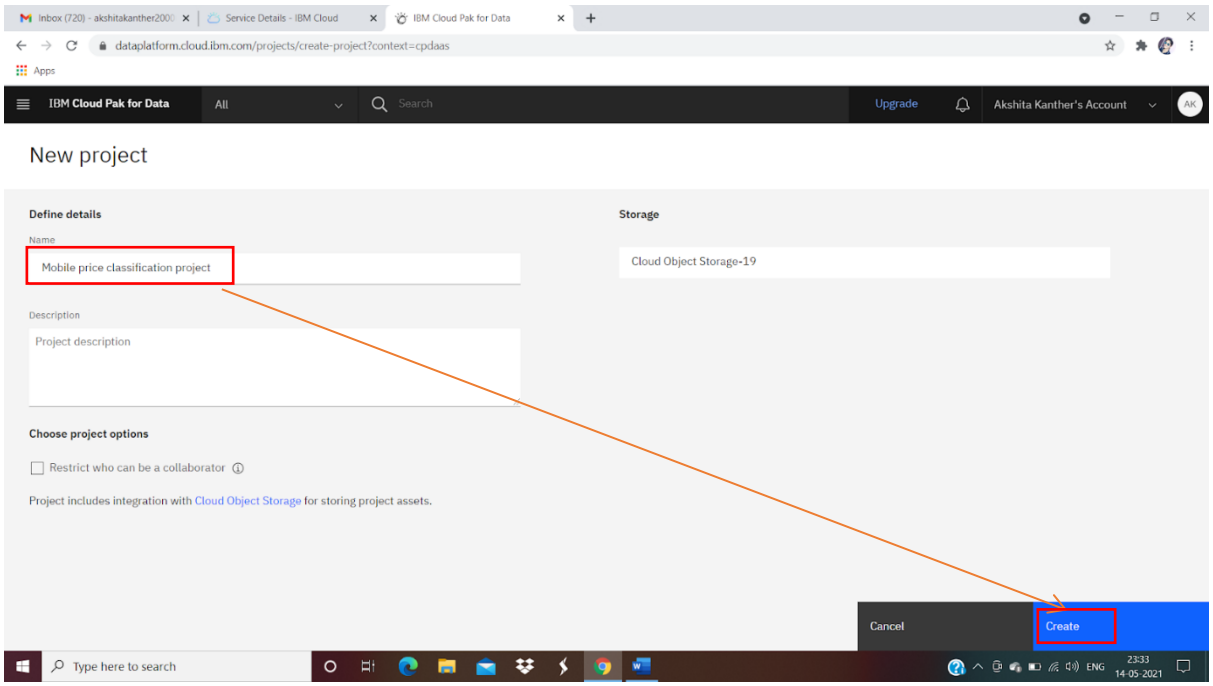
Step-4. Click on Create a project.



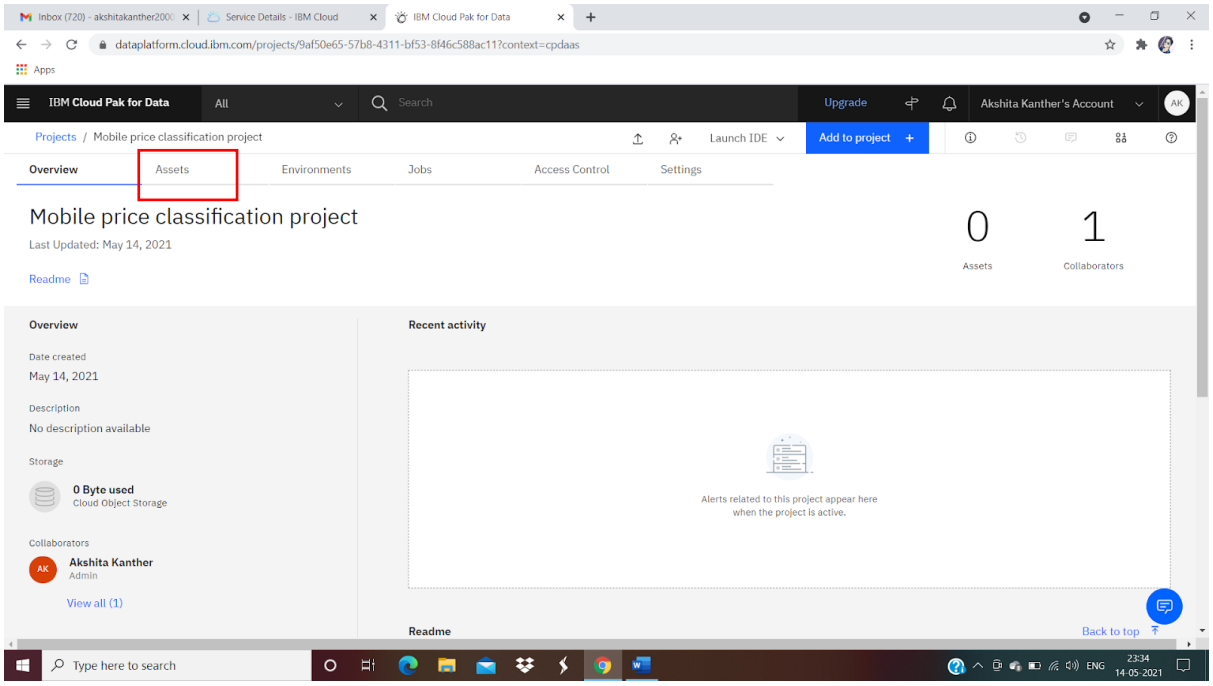
Step-5. Click on create an empty project.



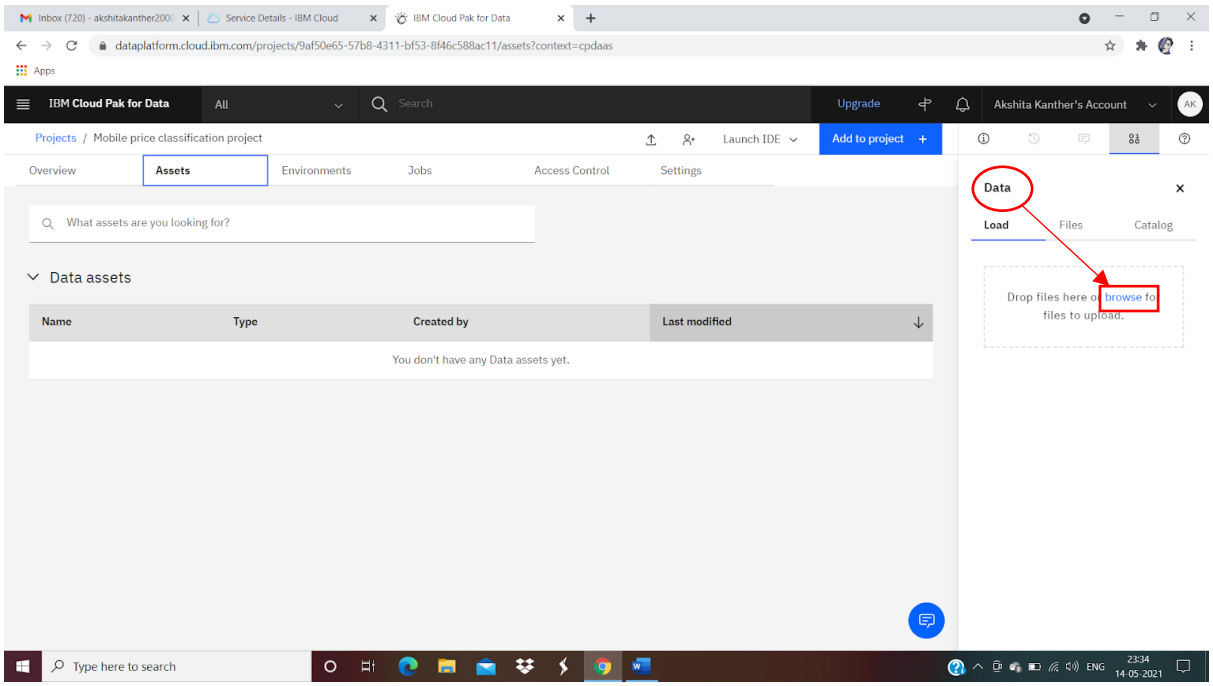
Step-6. Enter your project name. Also ensure that an instance of Cloud Object Storage is created. Enter project description (optional). Then click on Create.



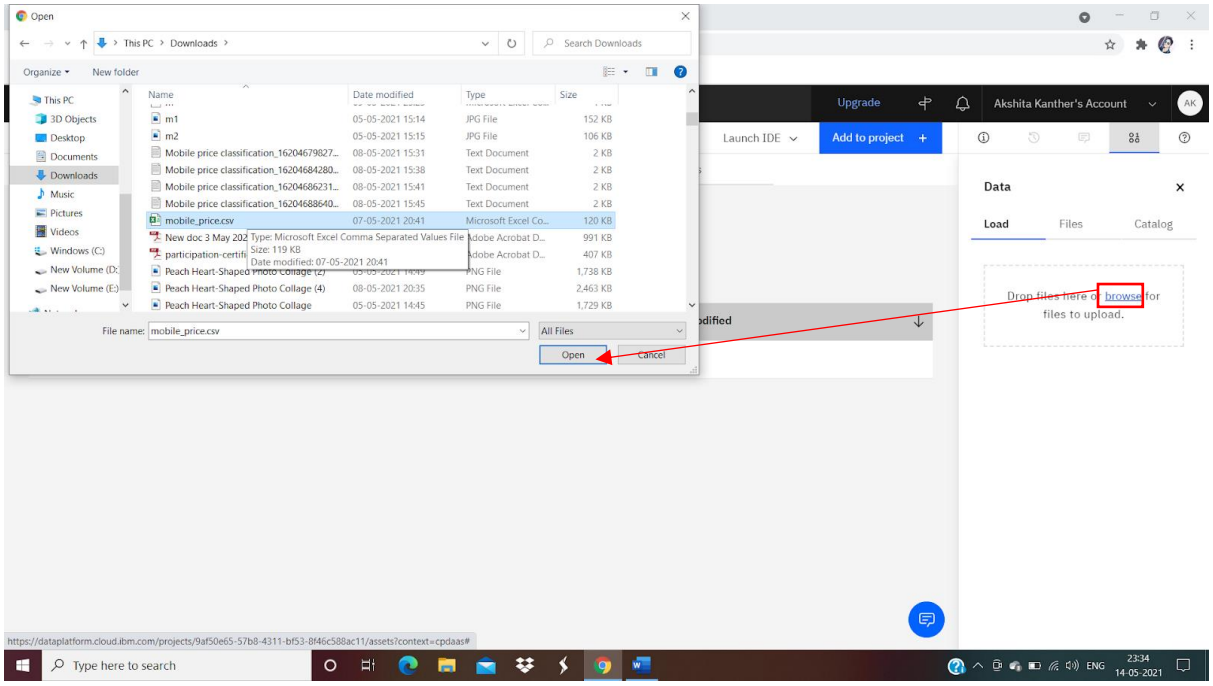
Step-7. Click on Assets.



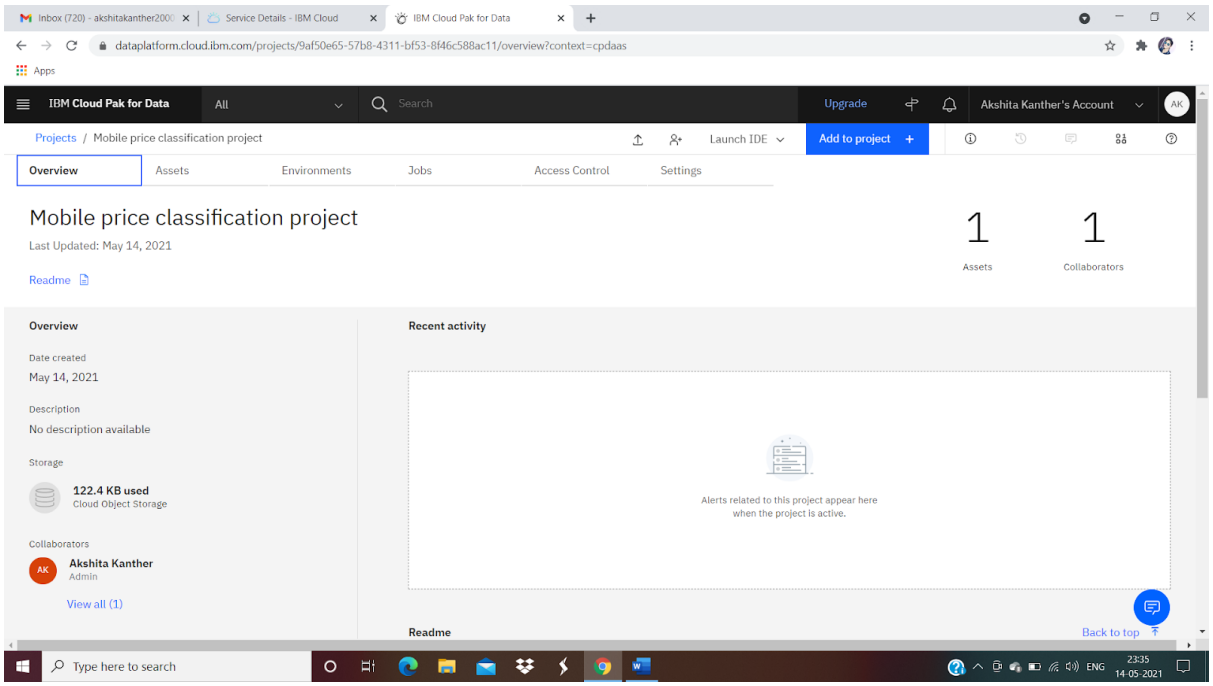
Step-8. Click on browse under load in the Data panel on the left side of the page.



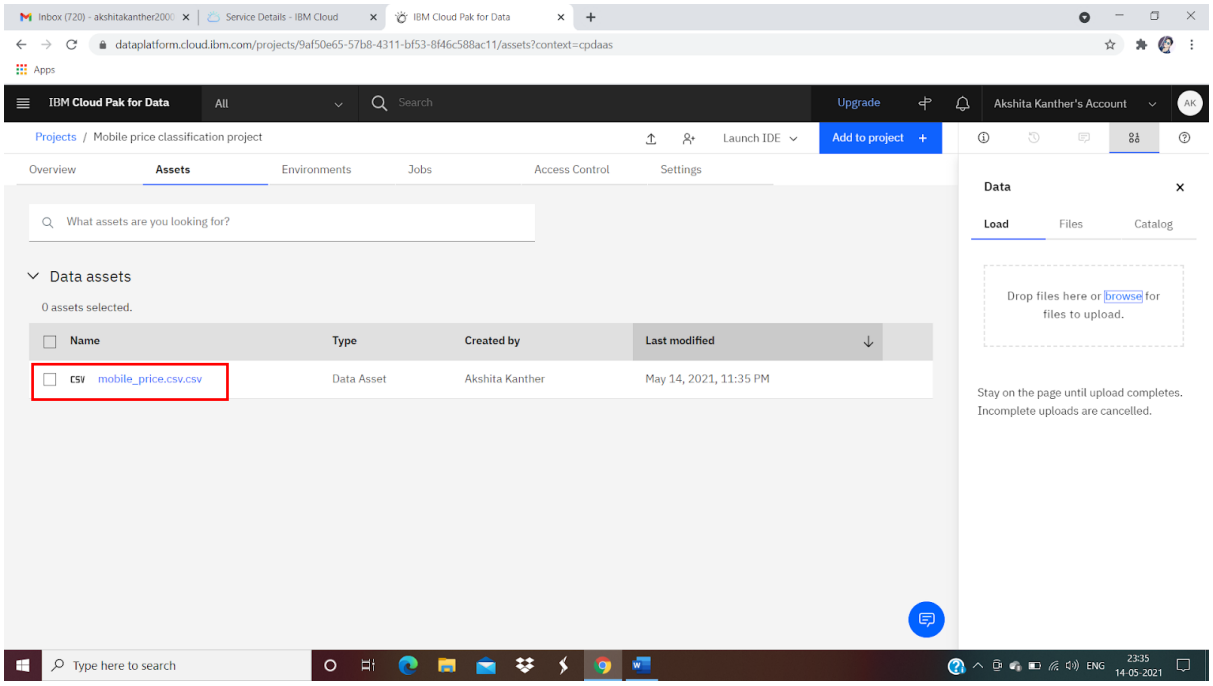
Step-9. And upload your dataset in CSV format.



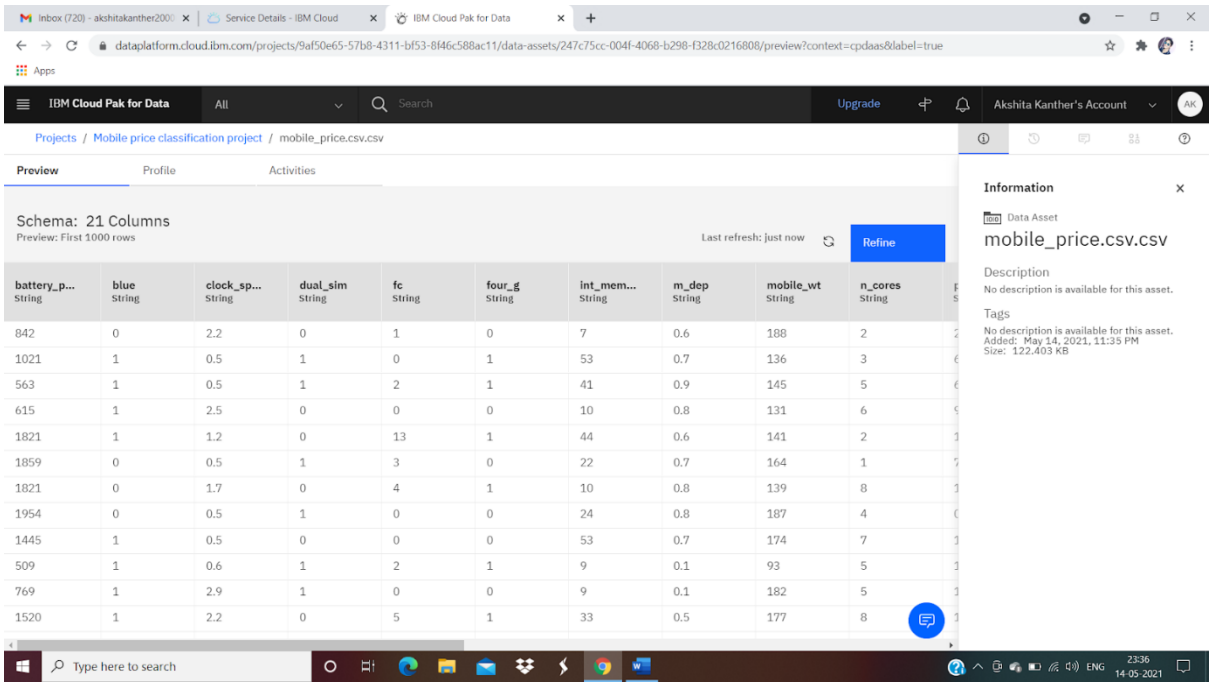
Step-10. Click on Overview to view that 1 Asset has been uploaded. Then switch to Assets.



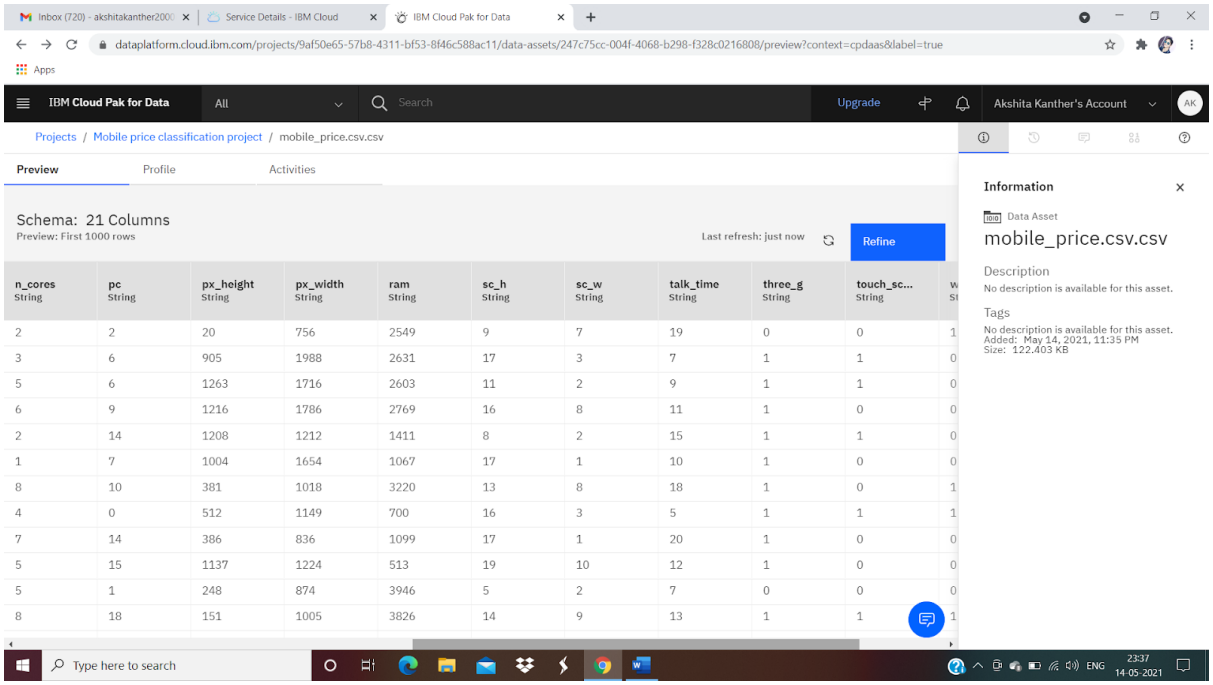
Step-11. Click on the name of csv file of dataset that has been uploaded.

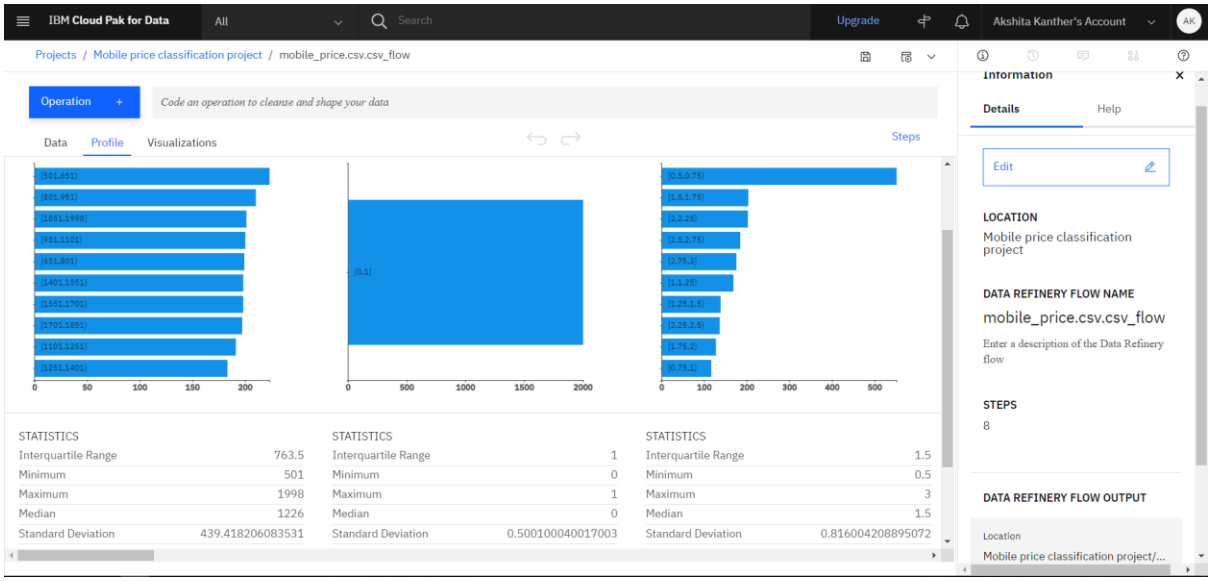


Step-12. Explore the dataset under preview.

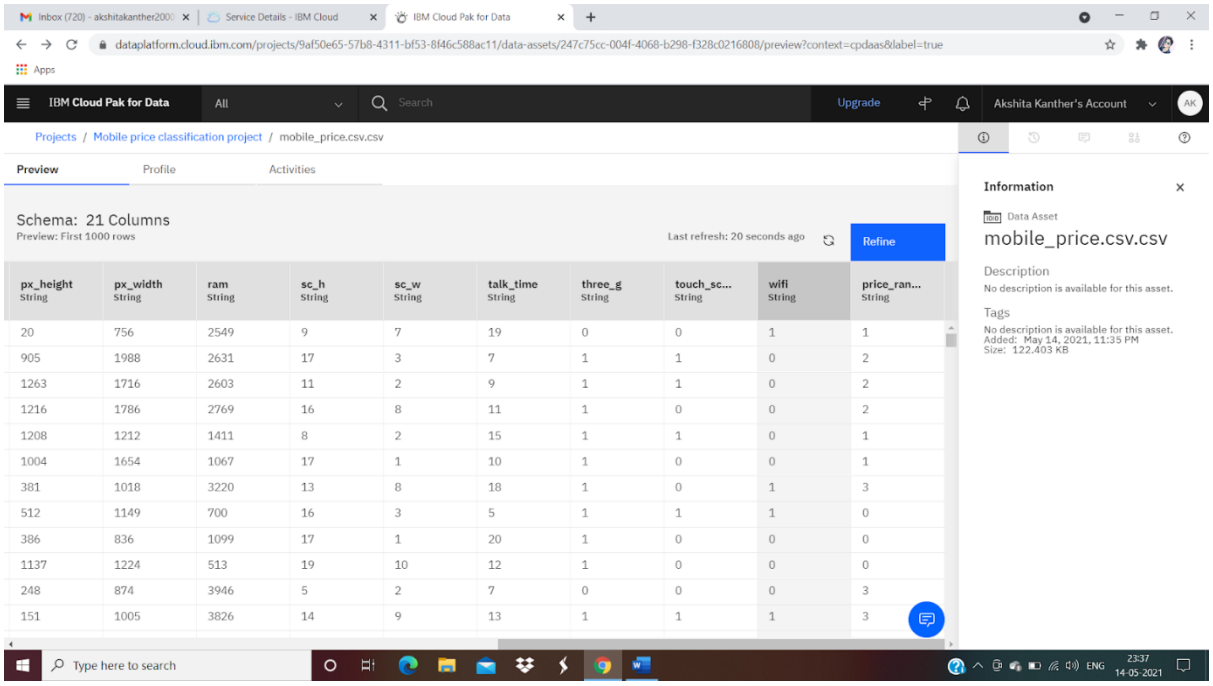


Step-13. Scroll to the right side and view all the columns.

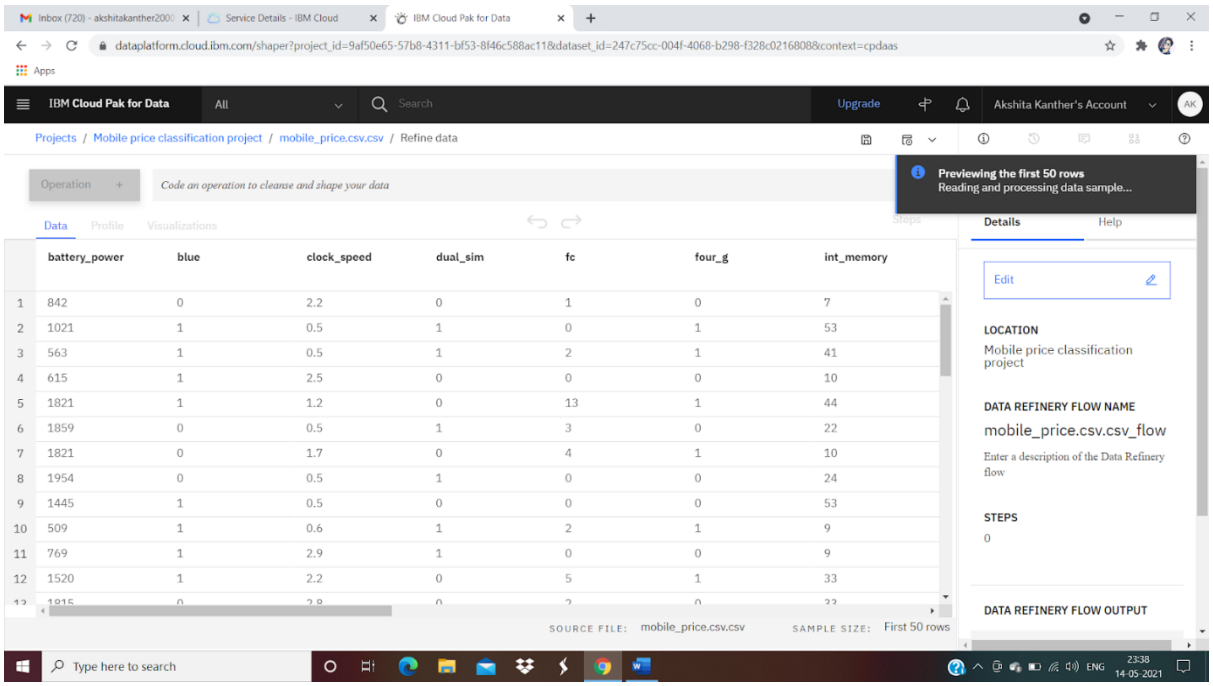




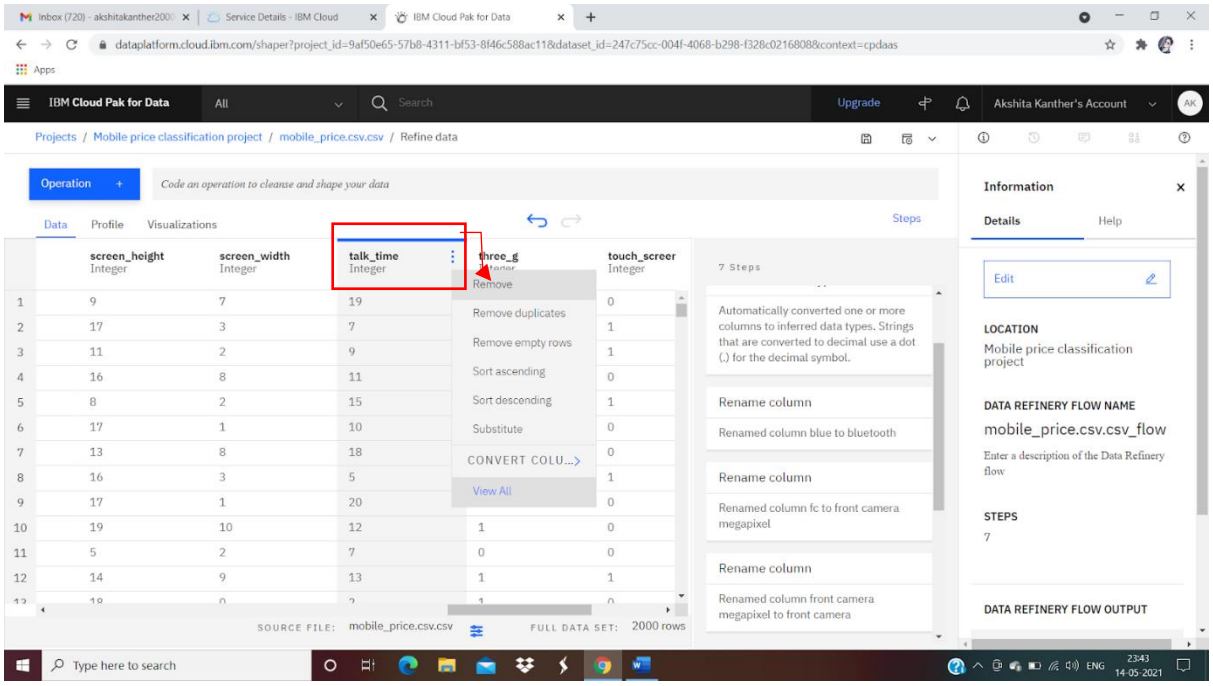
Step-14. To use Data refinery tools, Click on Refine.



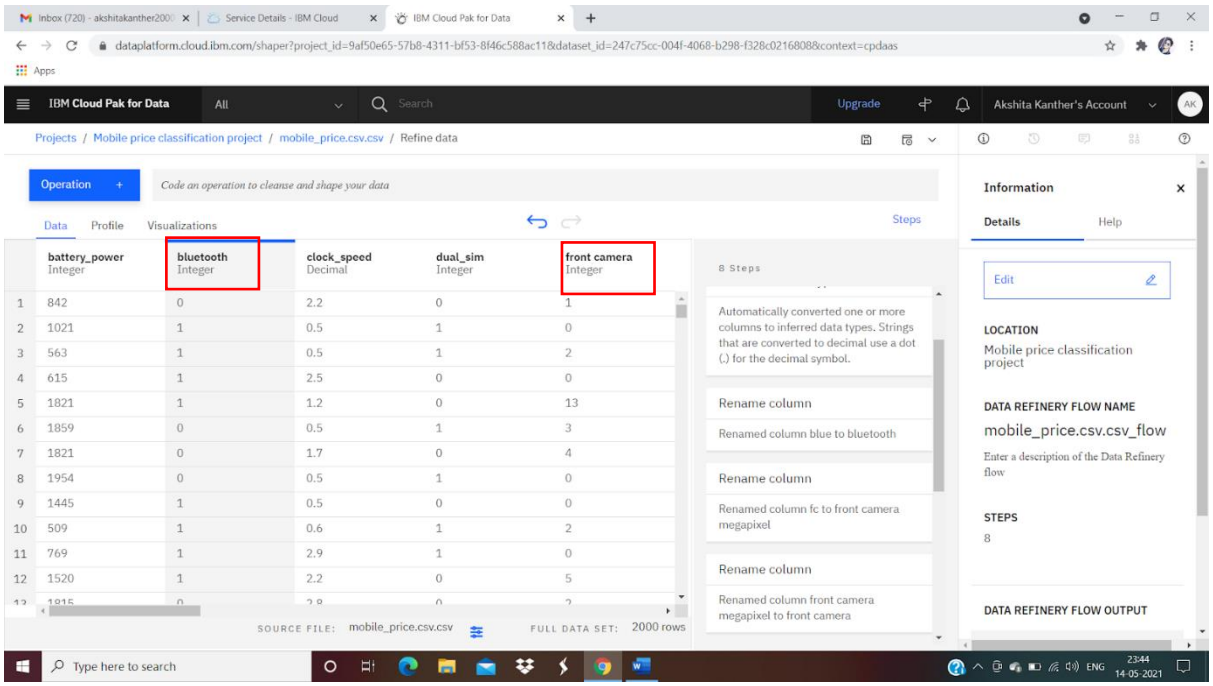
Step-15. You will find three tabs: Data, Profile and Visualizations. Click on Data.



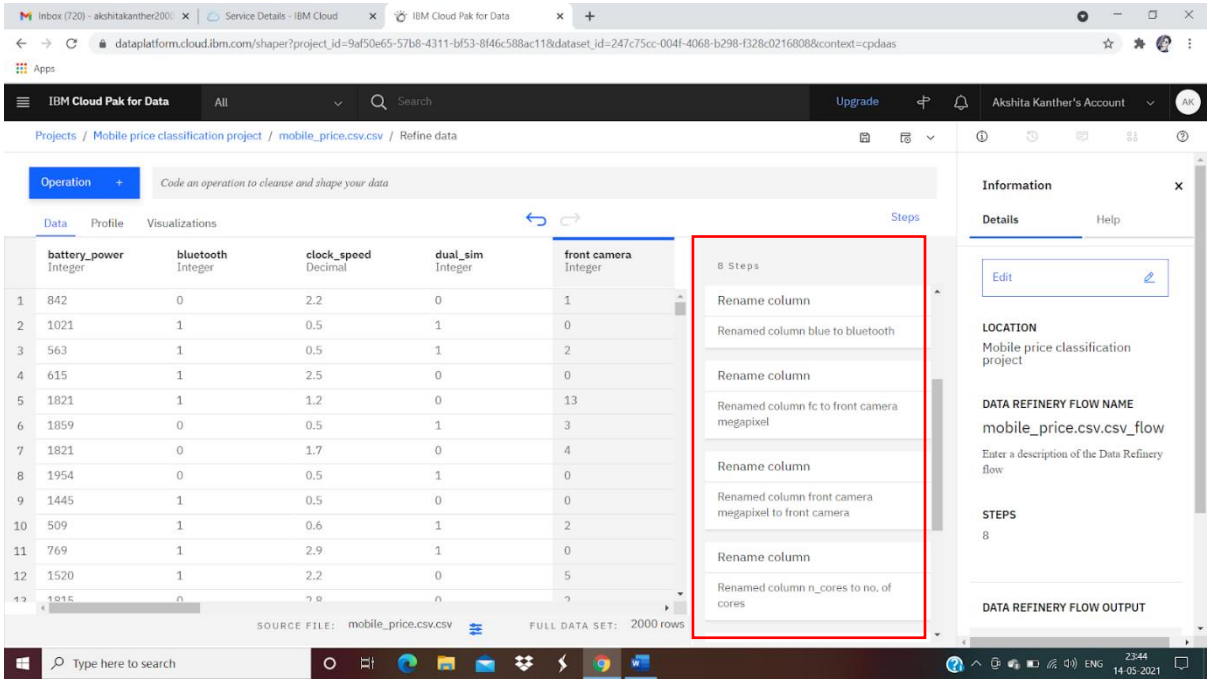
Step-16. Scroll to the right side and select the talk_time column, click on the three dots and Remove it as it is not required.



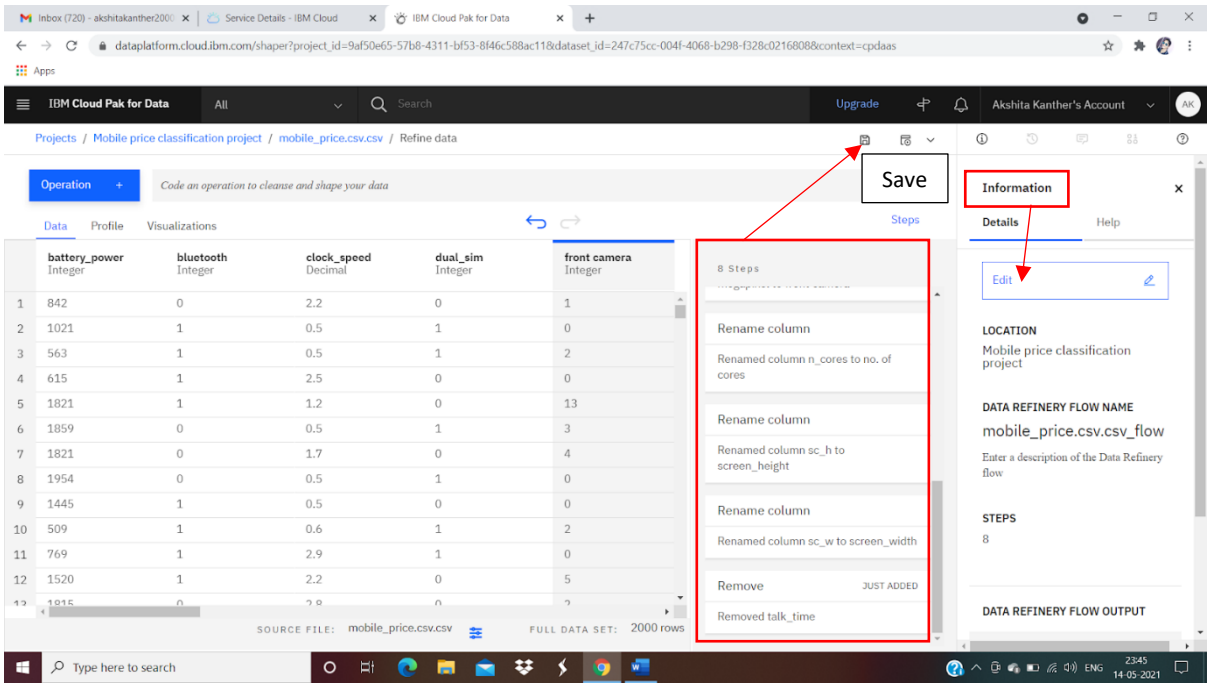
Step-17. Double click the header of blue column and rename it as Bluetooth, repeat the same for fc to rename it as front camera.



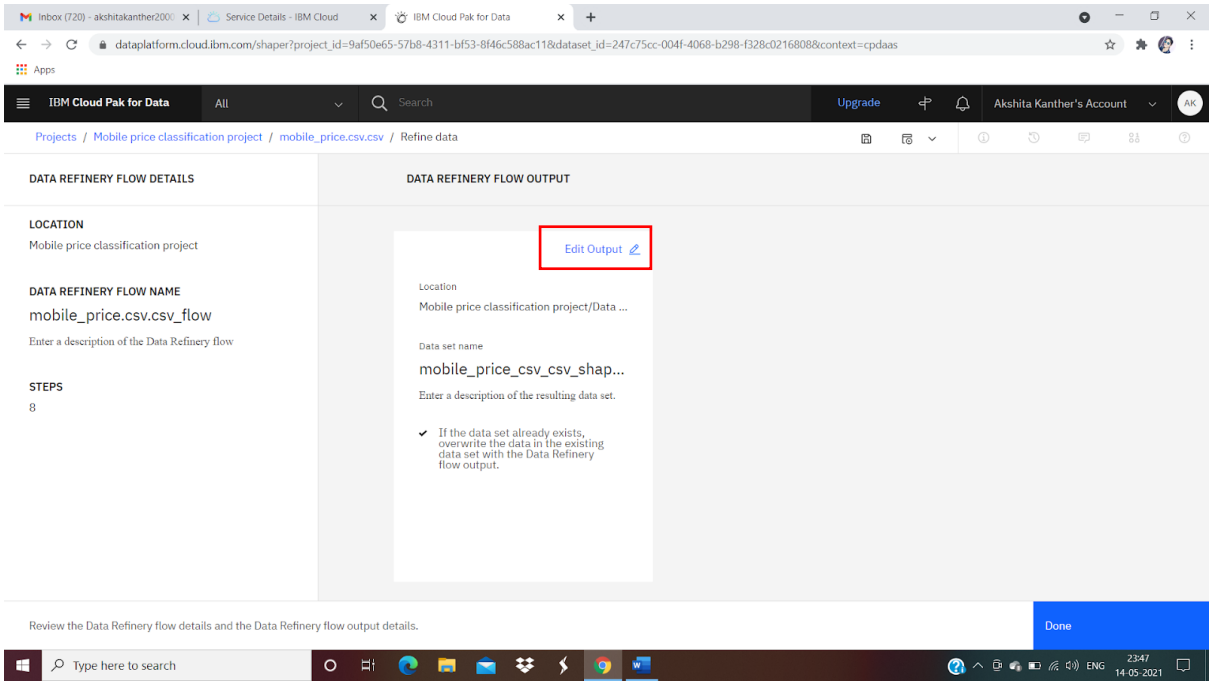
Step-18. Double click the header of n_cores column and rename it as no. of cores.



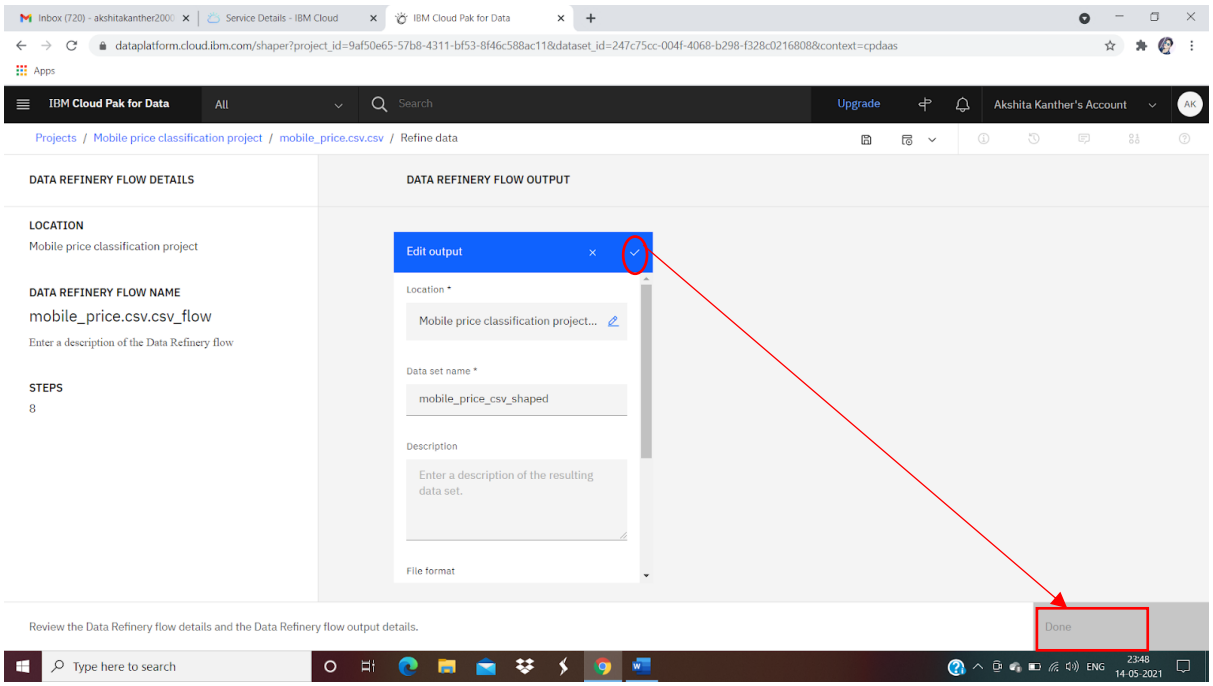
Step-19. Double click the header of sc_w column and rename it as screen_width. Then click on save icon on the top, then click on Edit on the right side under Details in Information pane.

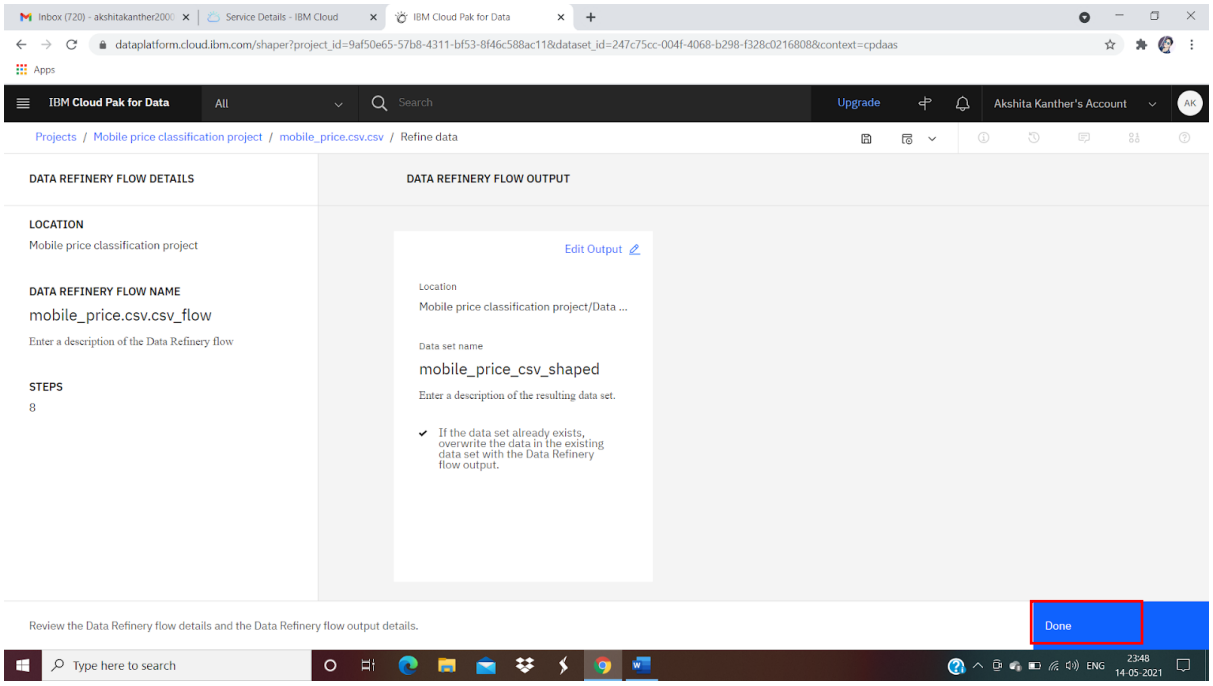


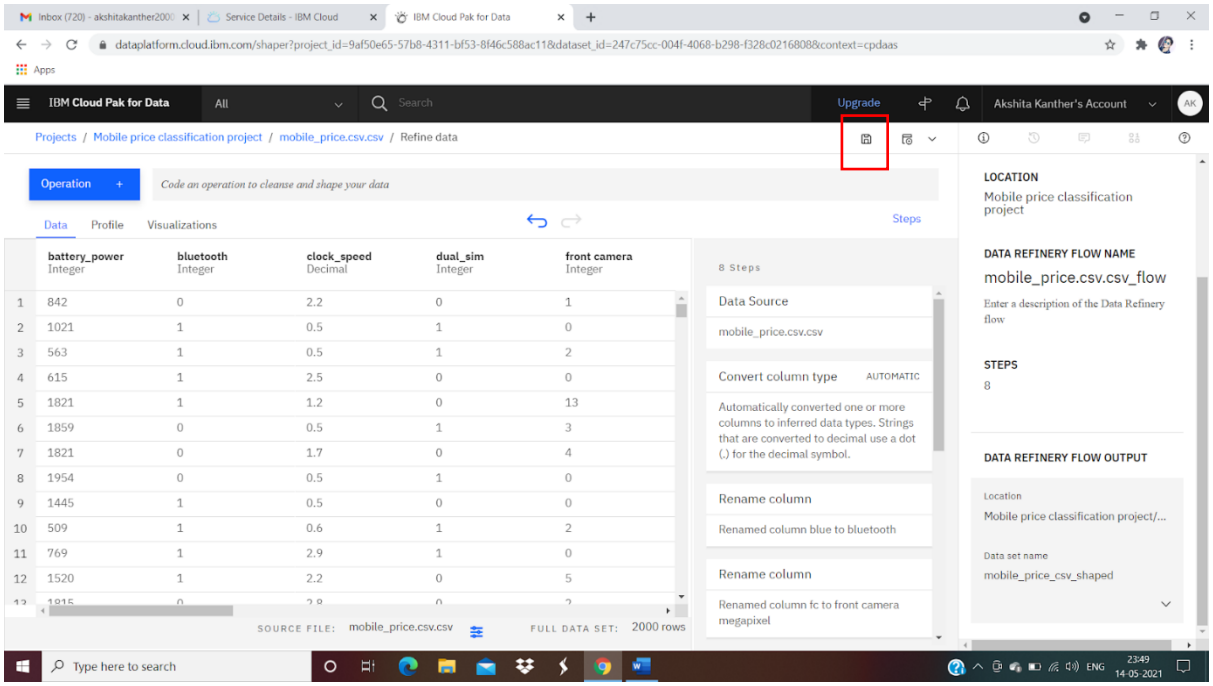
Step-20. Click on Edit Output.



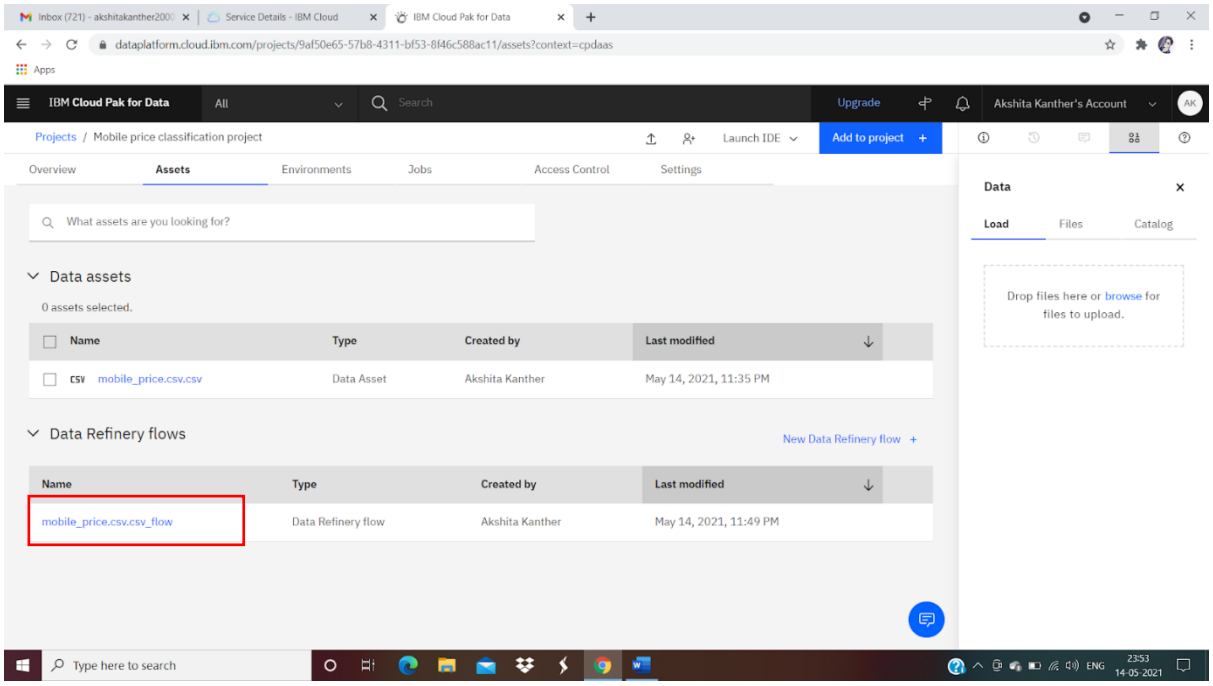
Step-21. Make the changes and click on Done and then click on save icon and save .





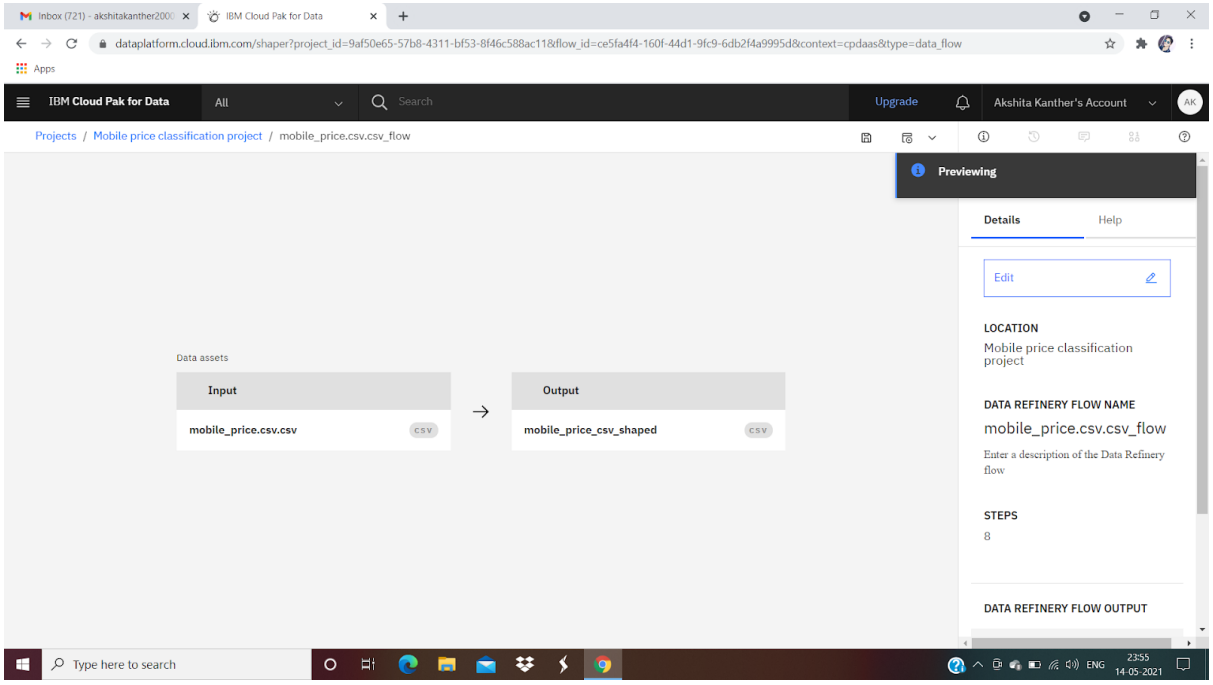


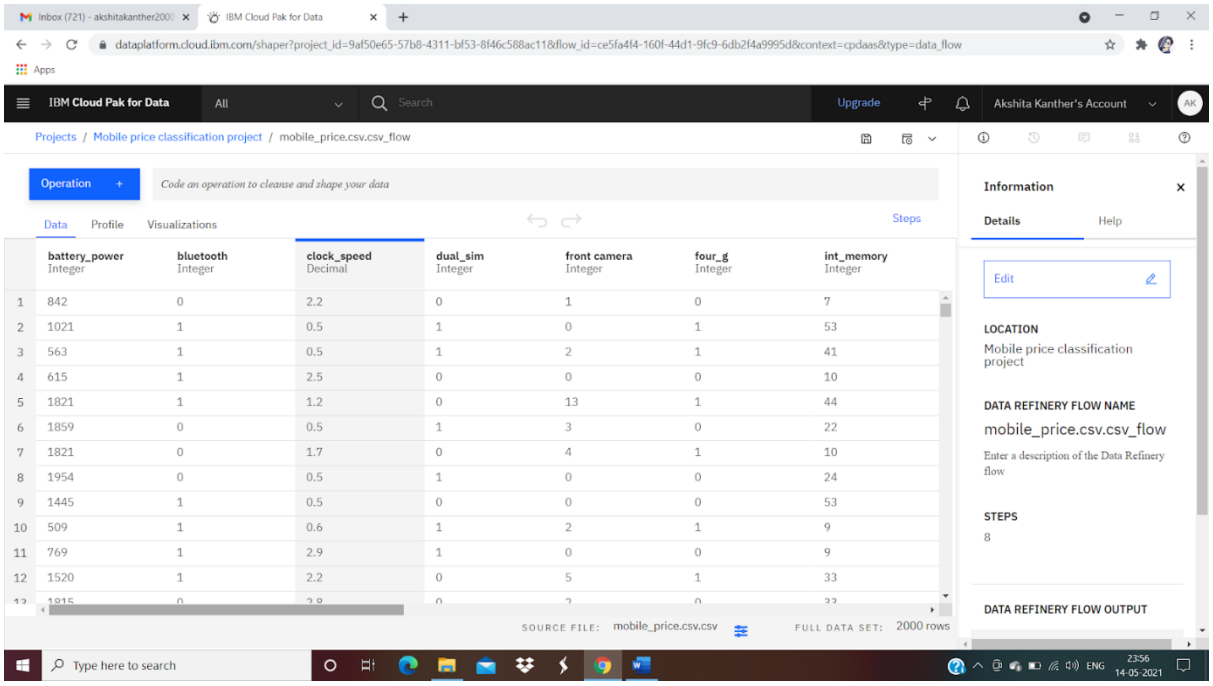
Step-22. Go to Assets and click on newly created Data refinery flow file under Data Refinery Flows.



After Refining

Data Refinery flow:-



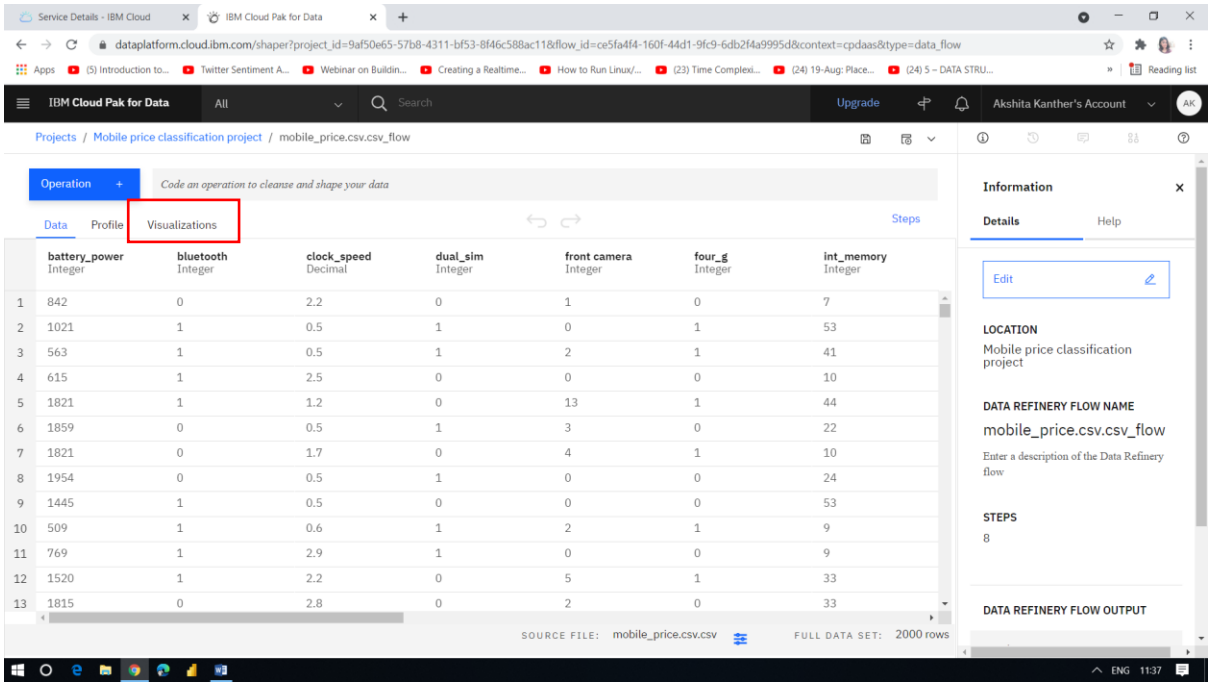


	battery_power Integer	bluetooth Integer	clock_speed Decimal	dual_sim Integer	front camera Integer	four_g Integer	int_memory Integer
1	842	0	2.2	0	1	0	7
2	1021	1	0.5	1	0	1	53
3	563	1	0.5	1	2	1	41
4	615	1	2.5	0	0	0	10
5	1821	1	1.2	0	13	1	44
6	1859	0	0.5	1	3	0	22
7	1821	0	1.7	0	4	1	10
8	1954	0	0.5	1	0	0	24
9	1445	1	0.5	0	0	0	53
10	509	1	0.6	1	2	1	9
11	769	1	2.9	1	0	0	9
12	1520	1	2.2	0	5	1	33

DATA VISUALIZATION-

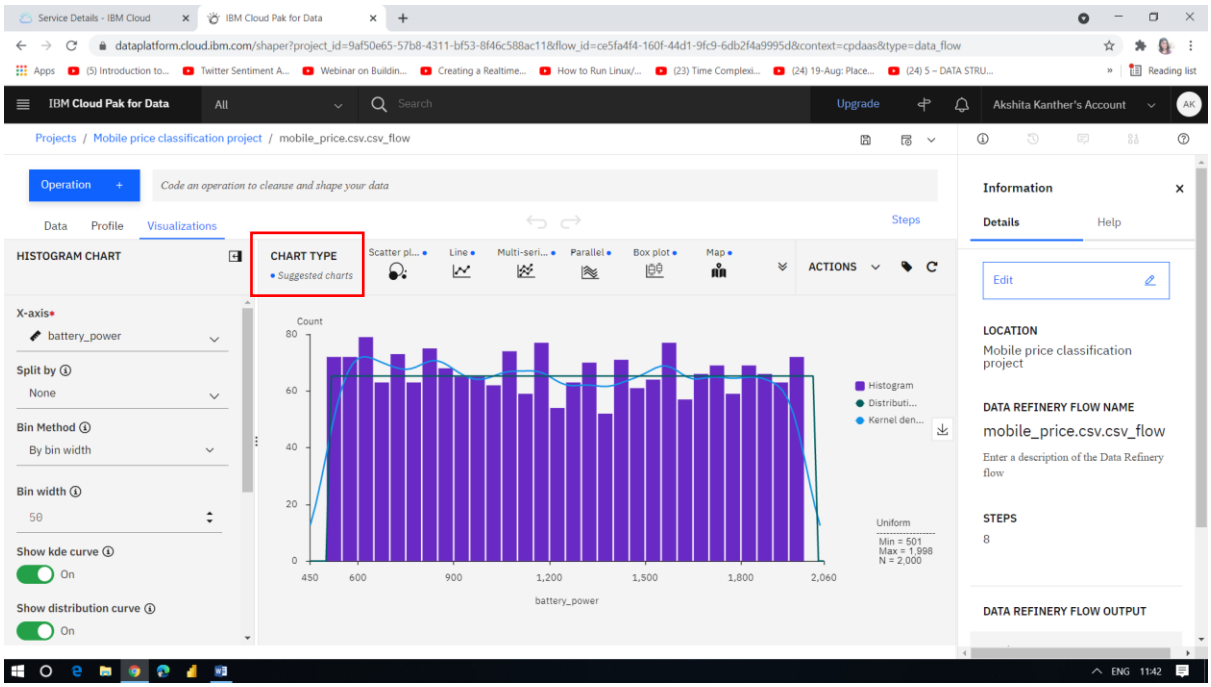
Data visualization is the representation of data through use of common graphics, such as charts, plots, infographics, and even animations. These visual displays of information communicate complex data relationships and data-driven insights in a way that is easy to understand.

Step-1. Click on visualizations tab.

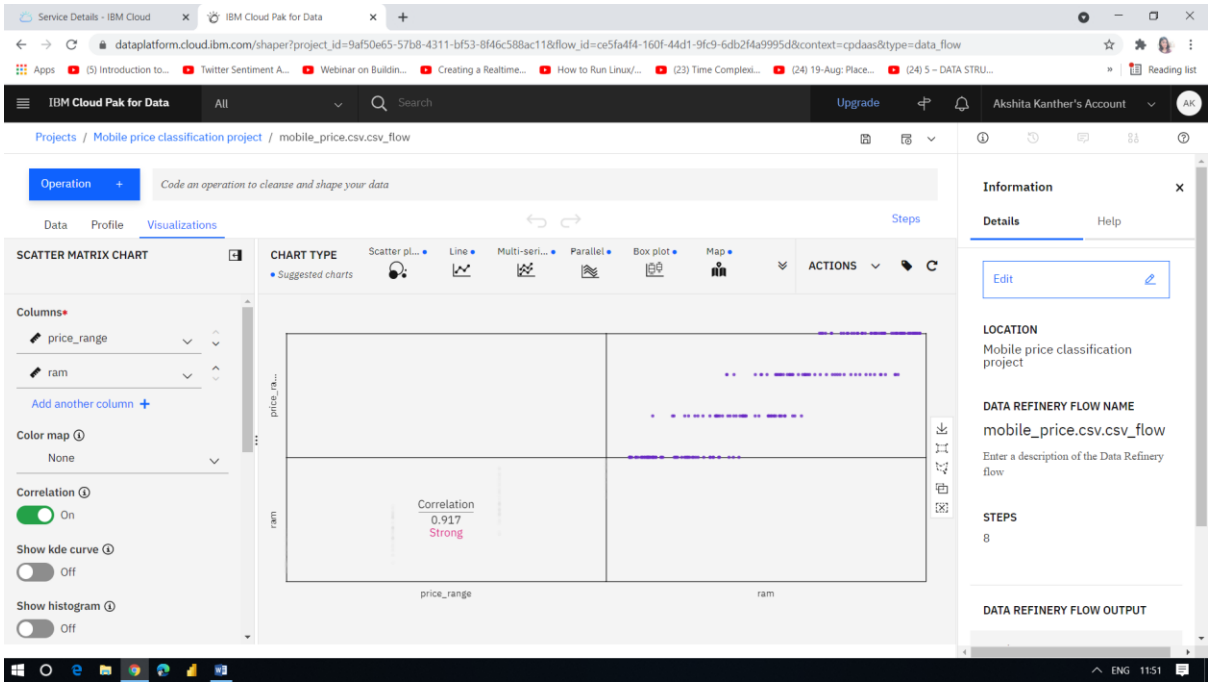


	battery_power Integer	bluetooth Integer	clock_speed Decimal	dual_sim Integer	front camera Integer	four_g Integer	int_memory Integer
1	842	0	2.2	0	1	0	7
2	1021	1	0.5	1	0	1	53
3	563	1	0.5	1	2	1	41
4	615	1	2.5	0	0	0	10
5	1821	1	1.2	0	13	1	44
6	1859	0	0.5	1	3	0	22
7	1821	0	1.7	0	4	1	10
8	1954	0	0.5	1	0	0	24
9	1445	1	0.5	0	0	0	53
10	509	1	0.6	1	2	1	9
11	769	1	2.9	1	0	0	9
12	1520	1	2.2	0	5	1	33
13	1815	0	2.8	0	2	0	33

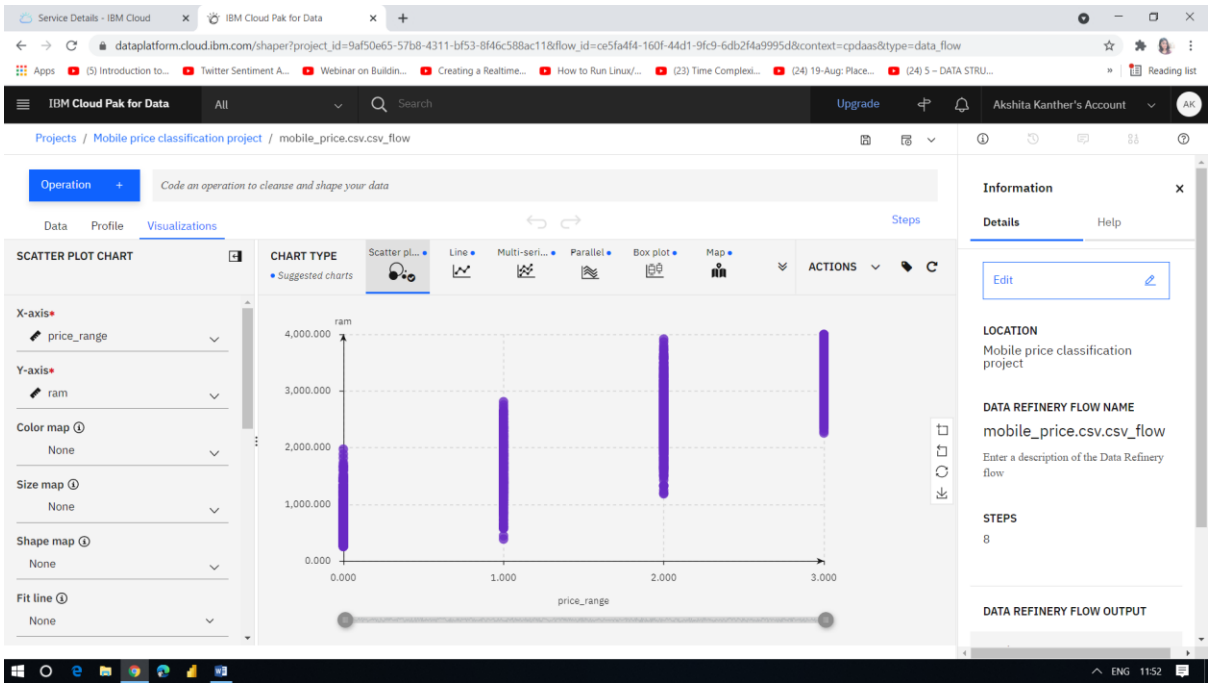
Step-2. Select Histogram chart, select battery_power for x-axis and bin method as Bin width. View the distribution for battery_power column.



Step-3. Select a SCATTER MATRIX CHART from the chart type and select price_range and ram as columns. We can observe strong correlation between them.



Step-4. Make a SCATTER PLOT CHART from chart type, select price_range for x-axis and select ram for y-axis.



VISUAL RECOGNITION SERVICE-

The IBM Watson Visual Recognition service uses deep learning algorithms to analyse images for scenes, objects, and other content. The response includes keywords that provide information about the content.

Available models

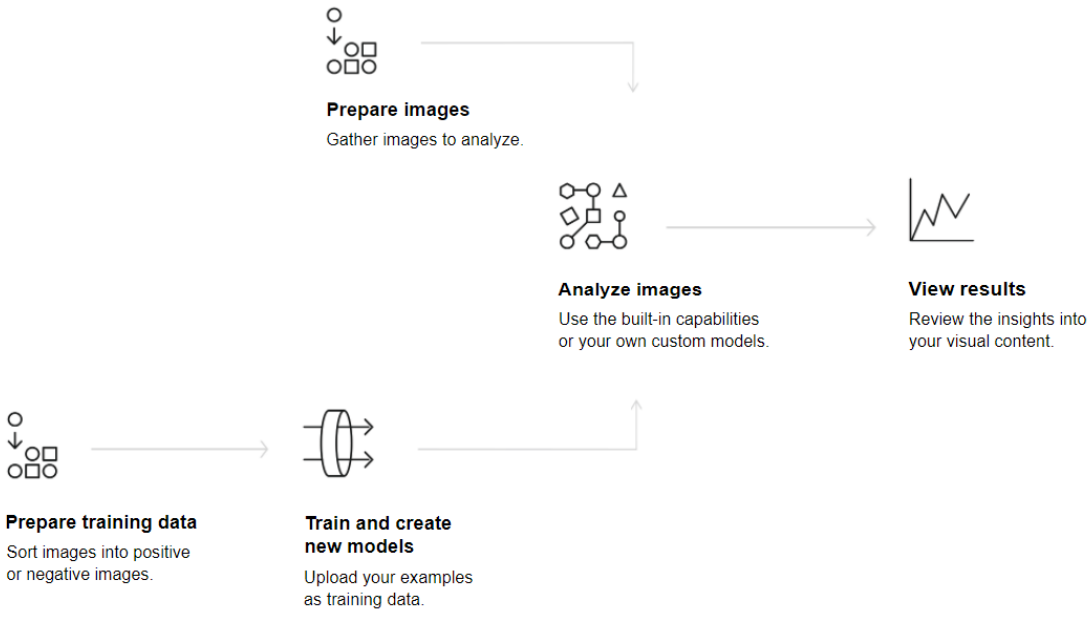
A set of built-in models provides highly accurate results without training:

- **General** model: Default classification from thousands of classes.
- **Explicit** model: Whether an image is inappropriate for general use.
- **Food** model: Specifically for images of food items.

We can also train **custom models** to create specialized classes.

How to use the service-

The following image shows the process of creating and using Visual Recognition:

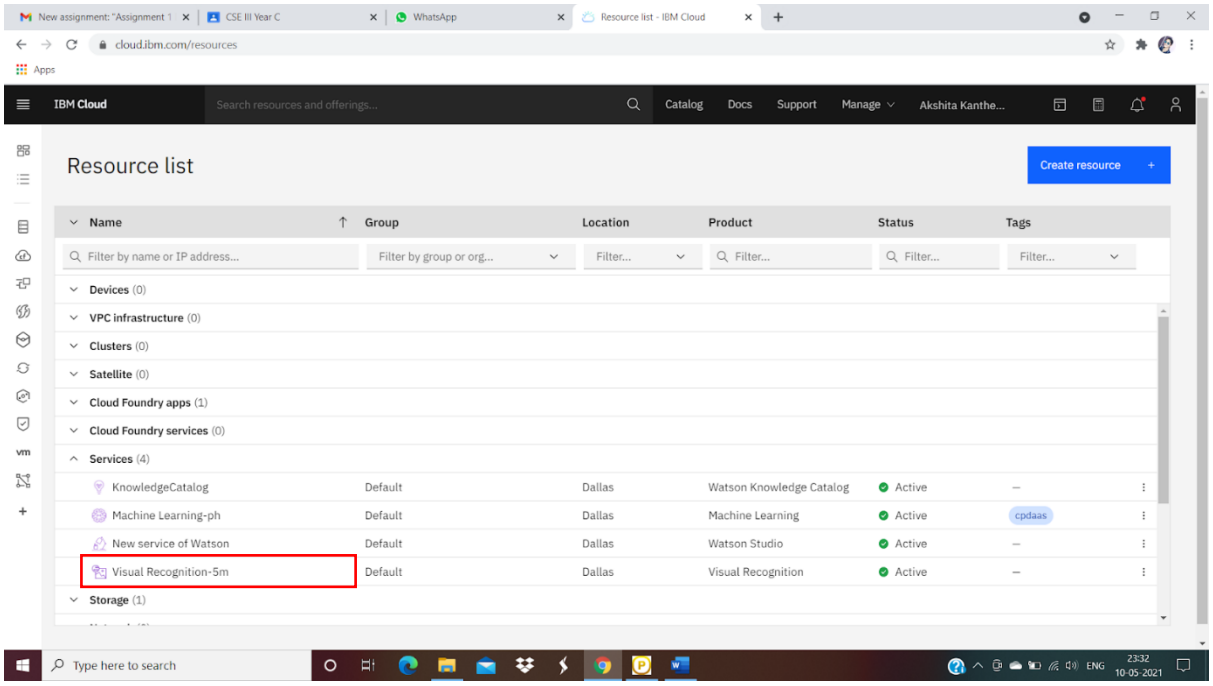


Use cases

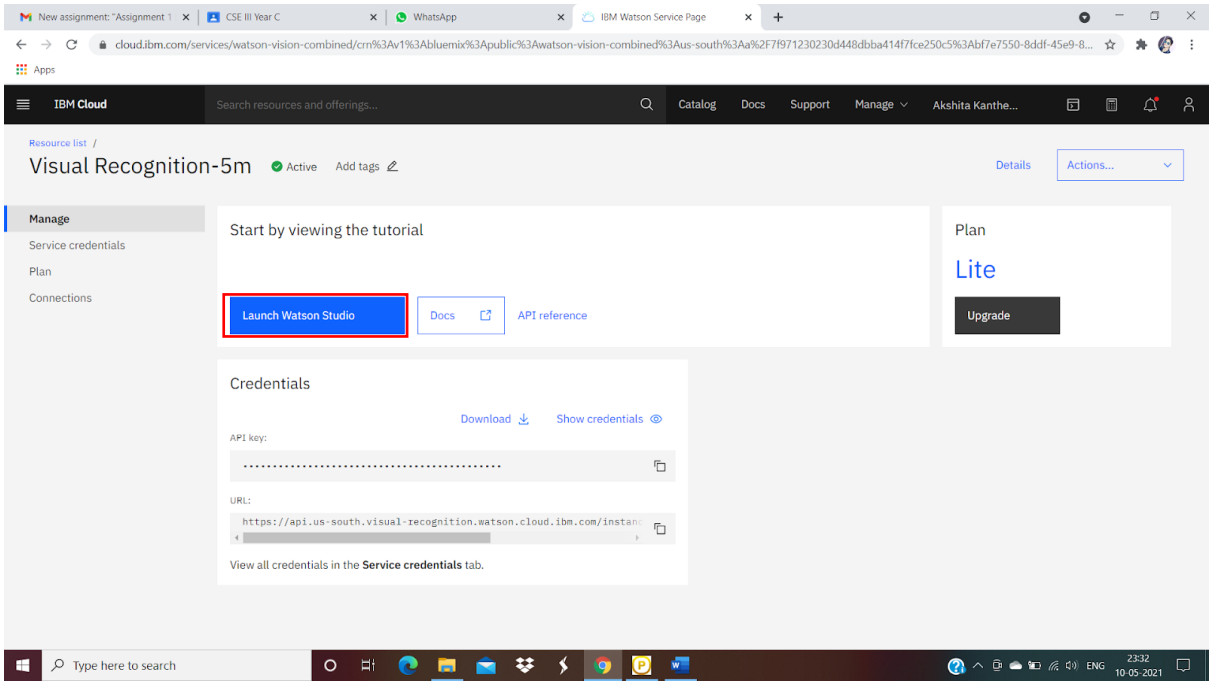
The Visual Recognition service can be used for diverse applications and industries, such as:

- **Manufacturing:** Use images from a manufacturing setting to make sure that products are positioned correctly on an assembly line
- **Visual auditing:** Look for visual compliance or deterioration in a fleet of trucks, planes, or windmills out in the field, train custom models to understand what defects look like
- **Insurance:** Rapidly process claims by using images to classify claims into different categories
- **Social listening:** Use images from your product line or your logo to track buzz about your company on social media
- **Social commerce:** Use an image of a plated dish to find out which restaurant serves it and find reviews, use a travel photo to find vacation suggestions based on similar experiences
- **Retail:** Take a photo of a favourite outfit to find stores with those clothes in stock or on sale, use a travel image to find retail suggestions in that area
- **Education:** Create image-based applications to educate about taxonomies

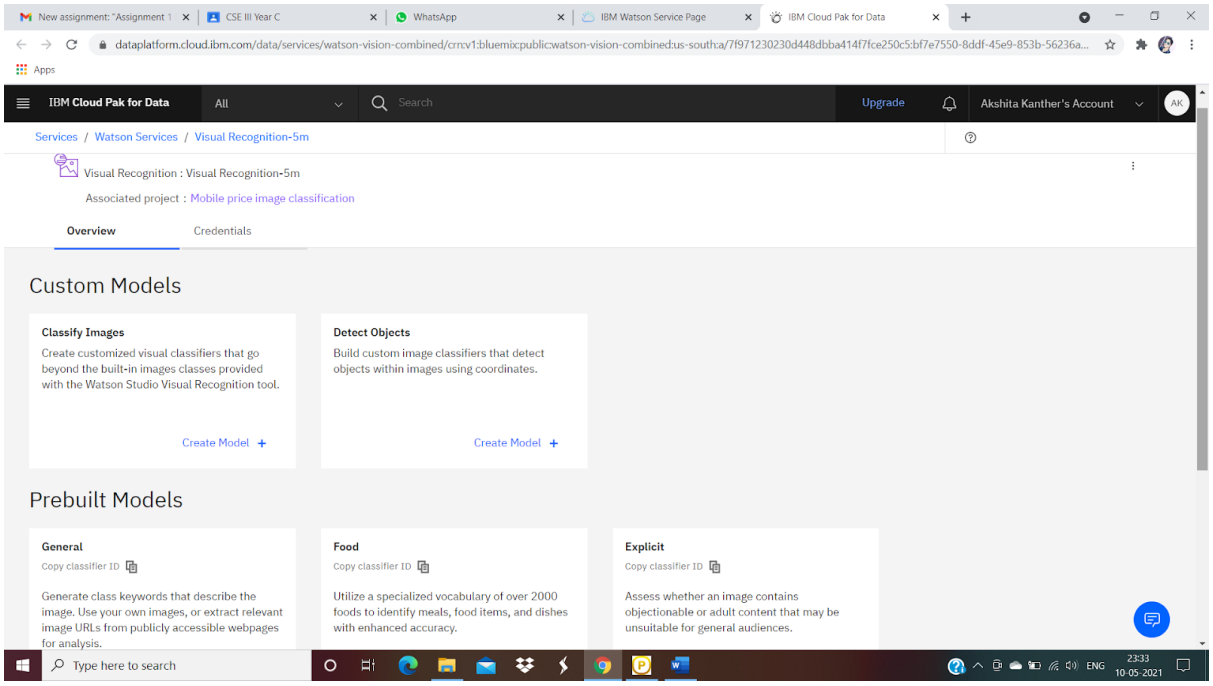
Step-1. Login to the IBM cloud platform, go to Resource list then click on Services and then click on instance of Visual Recognition Service. If an instance is not created previously then on the search bar of Search resources and offerings type visual recognition service and then create a visual recognition service instance.



Step-2. Click on Launch Watson Studio in Manage section.

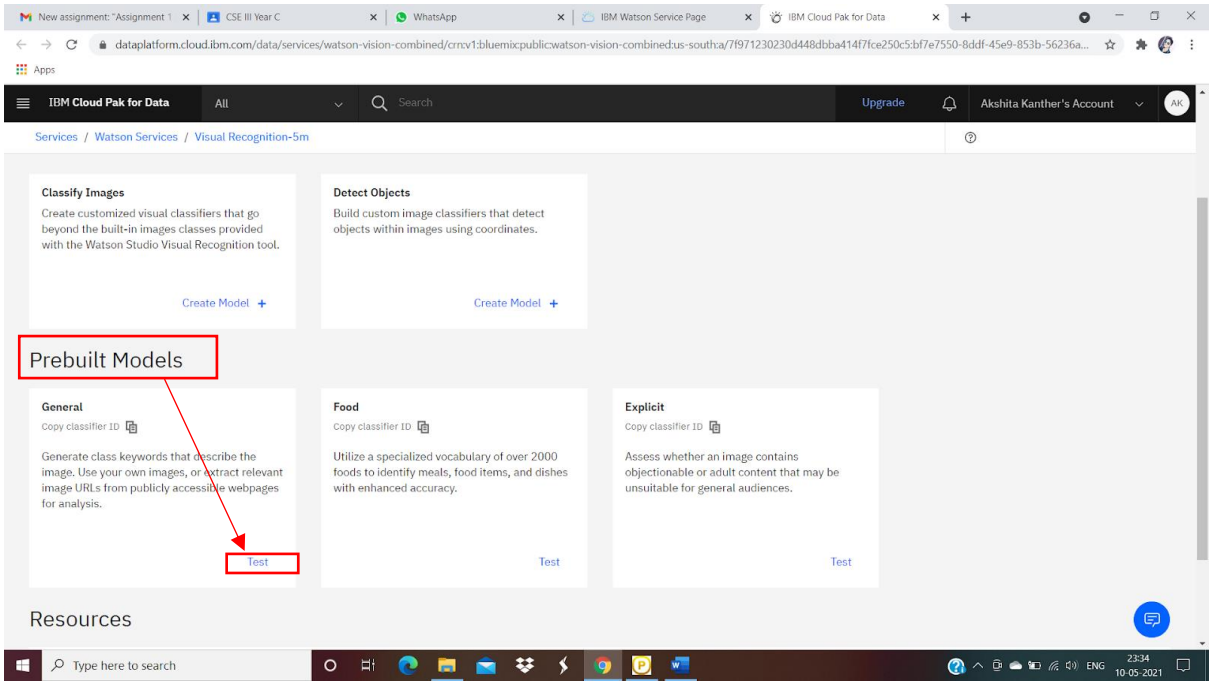


Step-3. We have options to create two types of models, one is Custom Models and another one is Prebuilt Models. We will be working on both types of models.

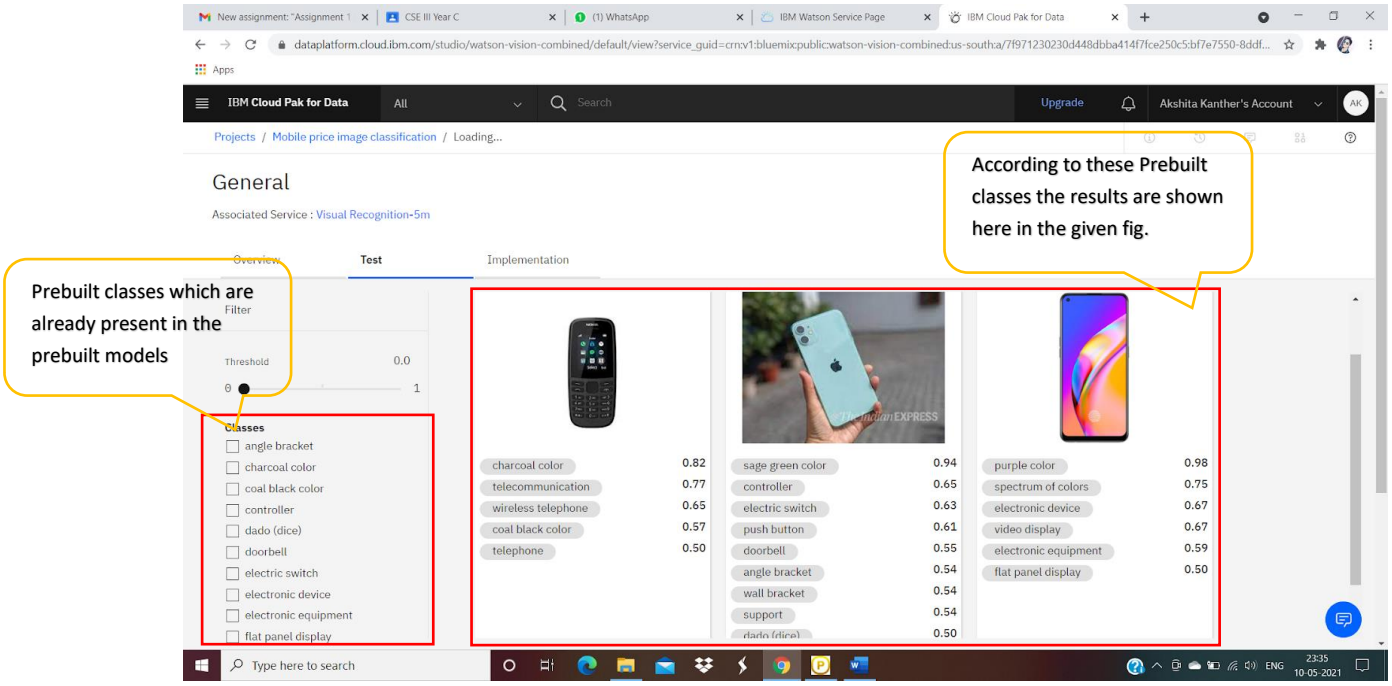


Working on Prebuilt models

Step-4. Go to General under Prebuilt Models and click on Test.

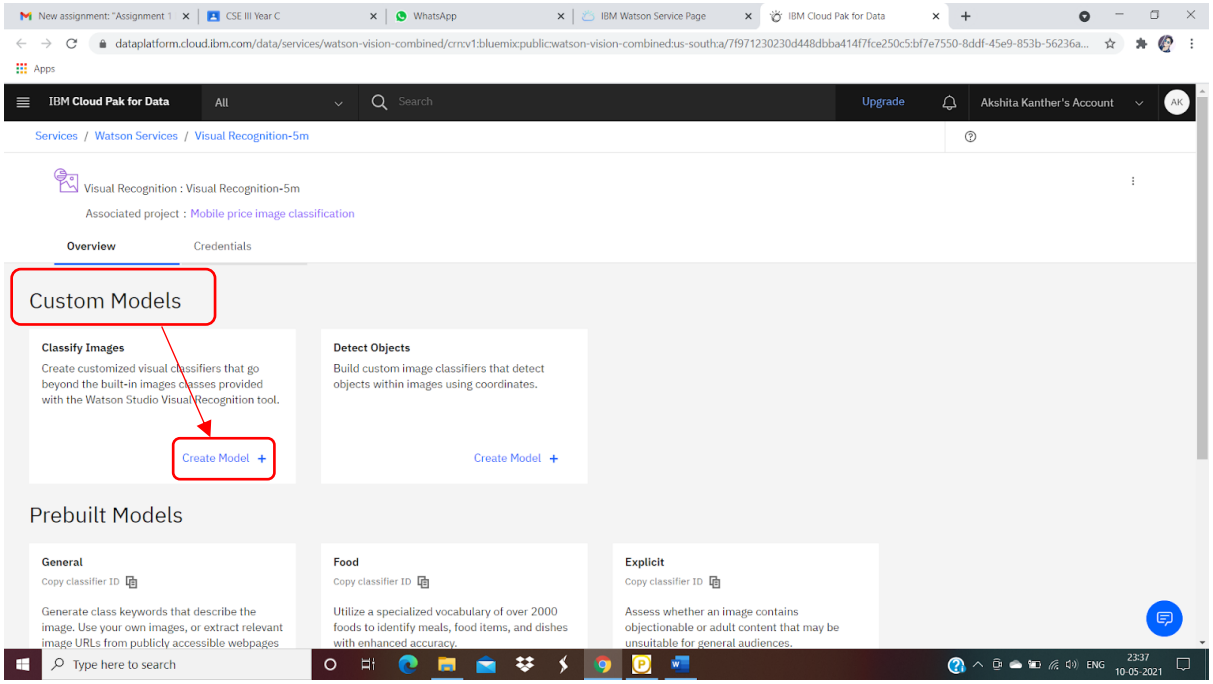


Step-5. Upload the images to Test and output will be displayed on the basis of multiple prebuilt classes.

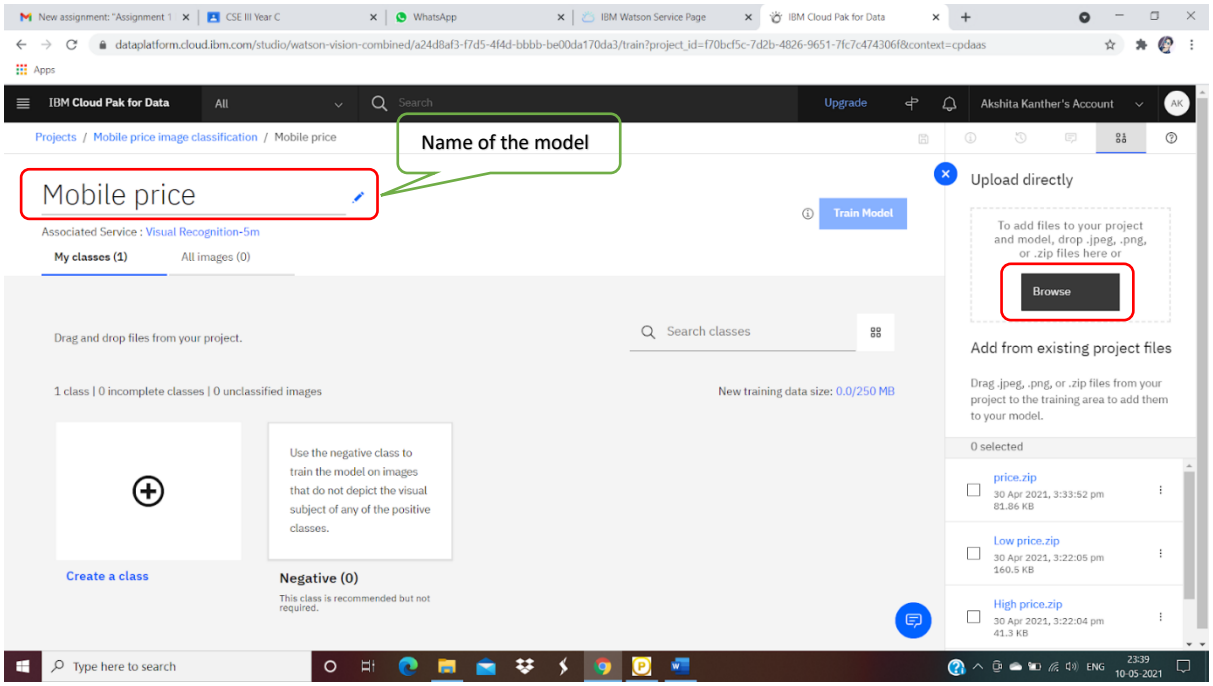


Working on Custom models

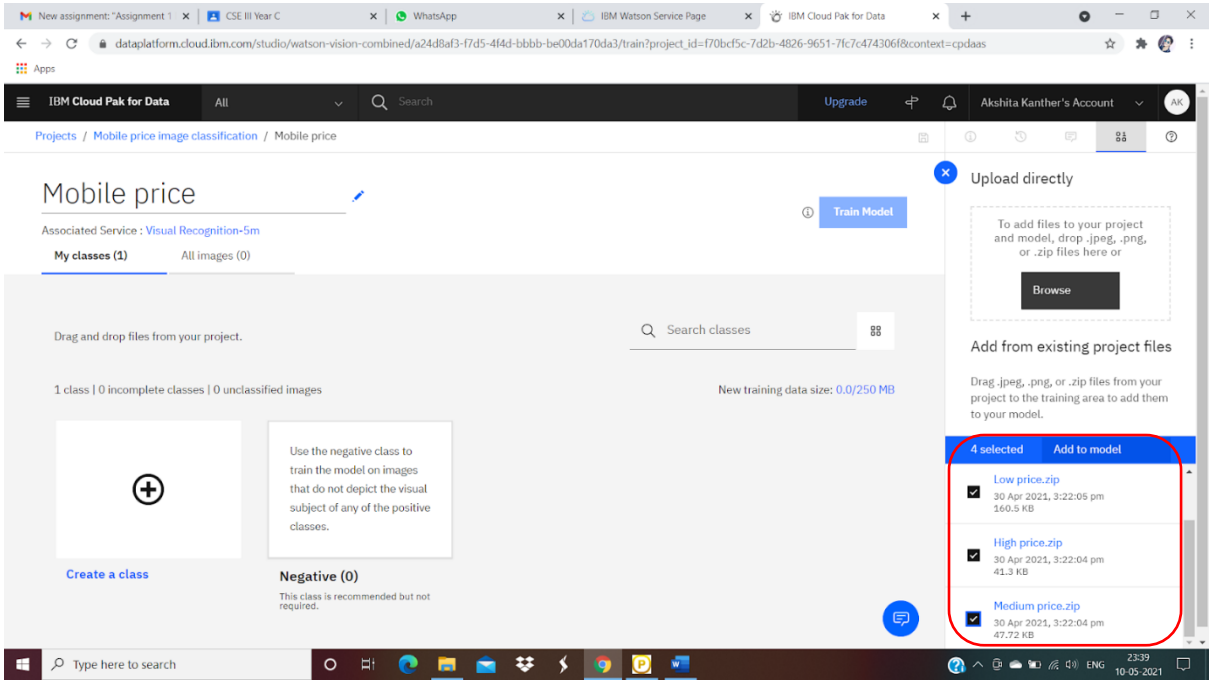
Step-6. Go to Classify Images in Custom Model and click on Create Model + .



Step-7. Go to the Upload directly section on the right and then click on Browse to upload image folders or zip files of images of different classes: High price, low price and medium price and also we can name the model.

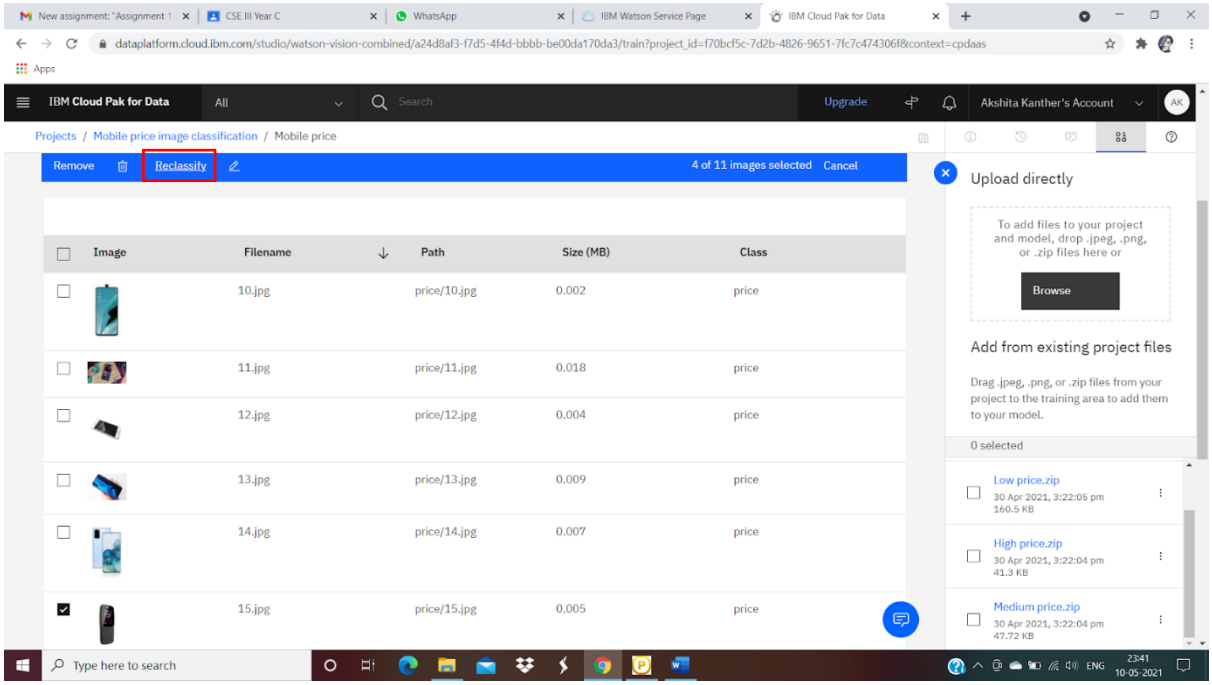


Step-8. Select all the image zip folders and then click on Add to model.

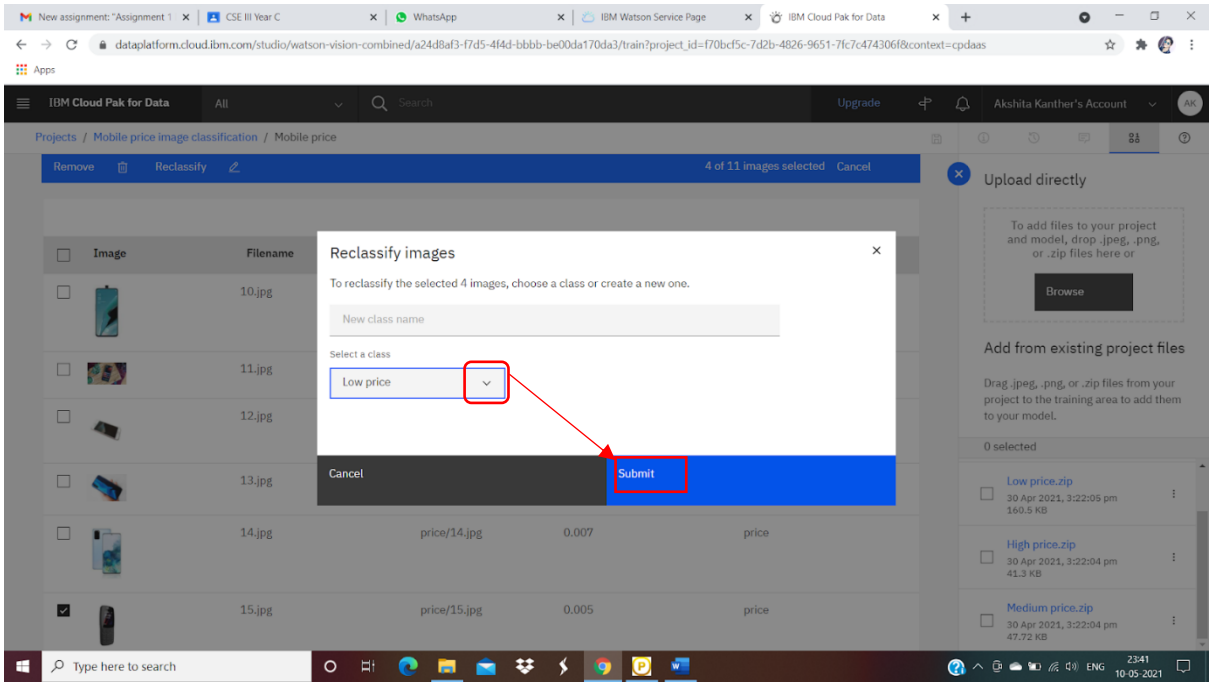


Reclassifying the images –

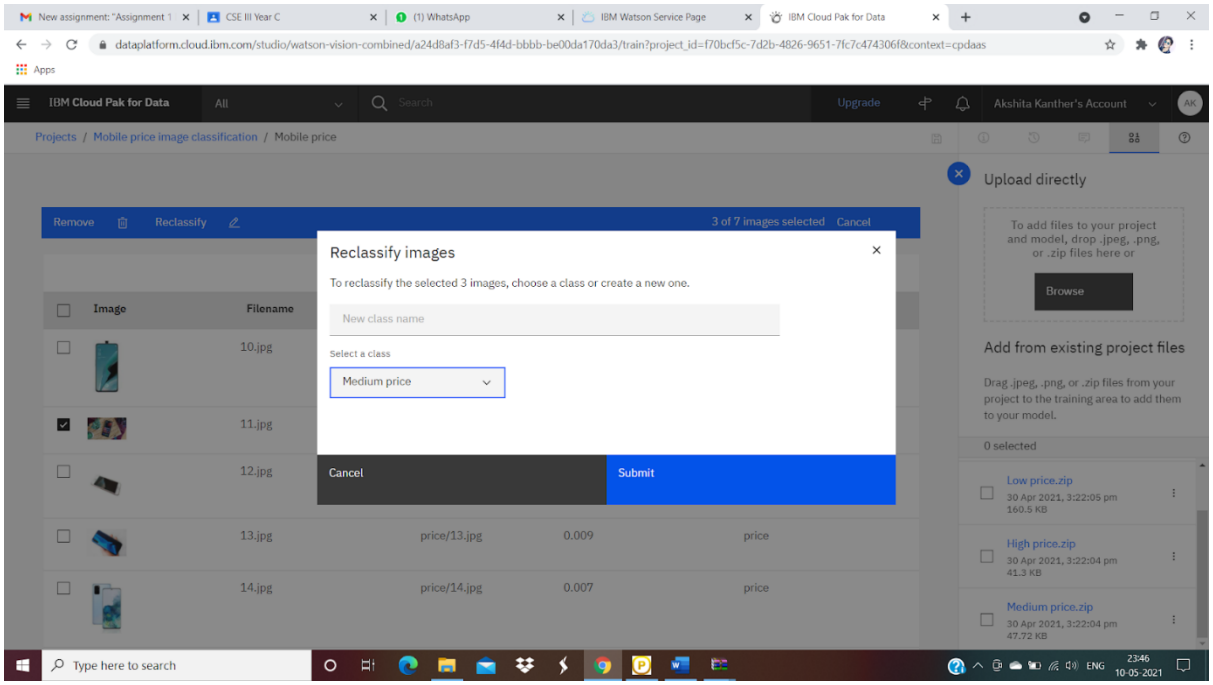
Step-9. Reclassify the images if they are in incorrect classes. Open each folder then select the images which shouldn't be present in that class, and then click on Reclassify on the top.

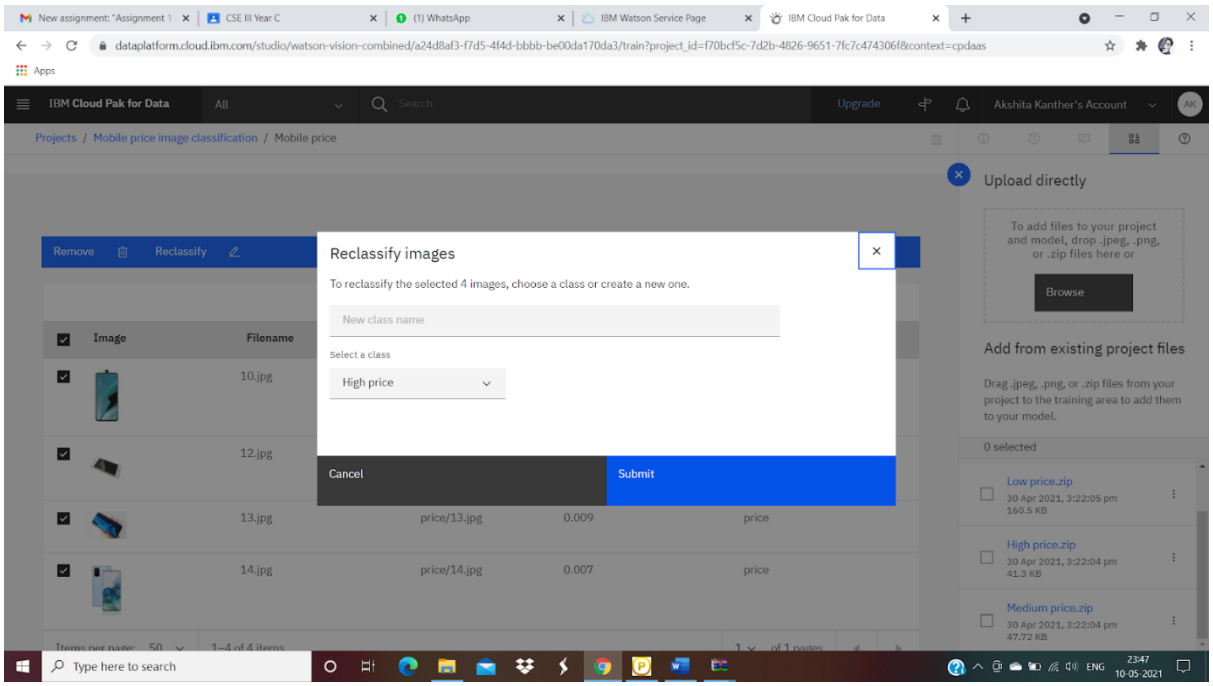


Step-10. A dialog box will pop up. Select an appropriate class for those selected items and then click on Submit.

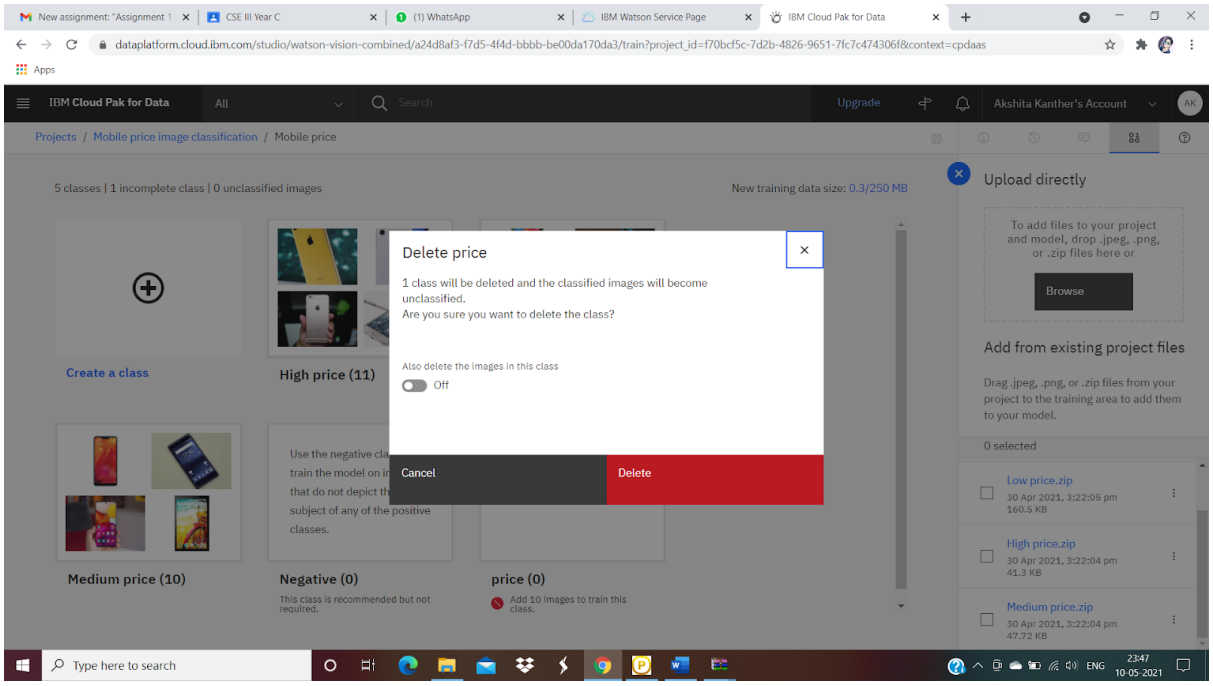


Step-11. Repeat the same for other classes also.

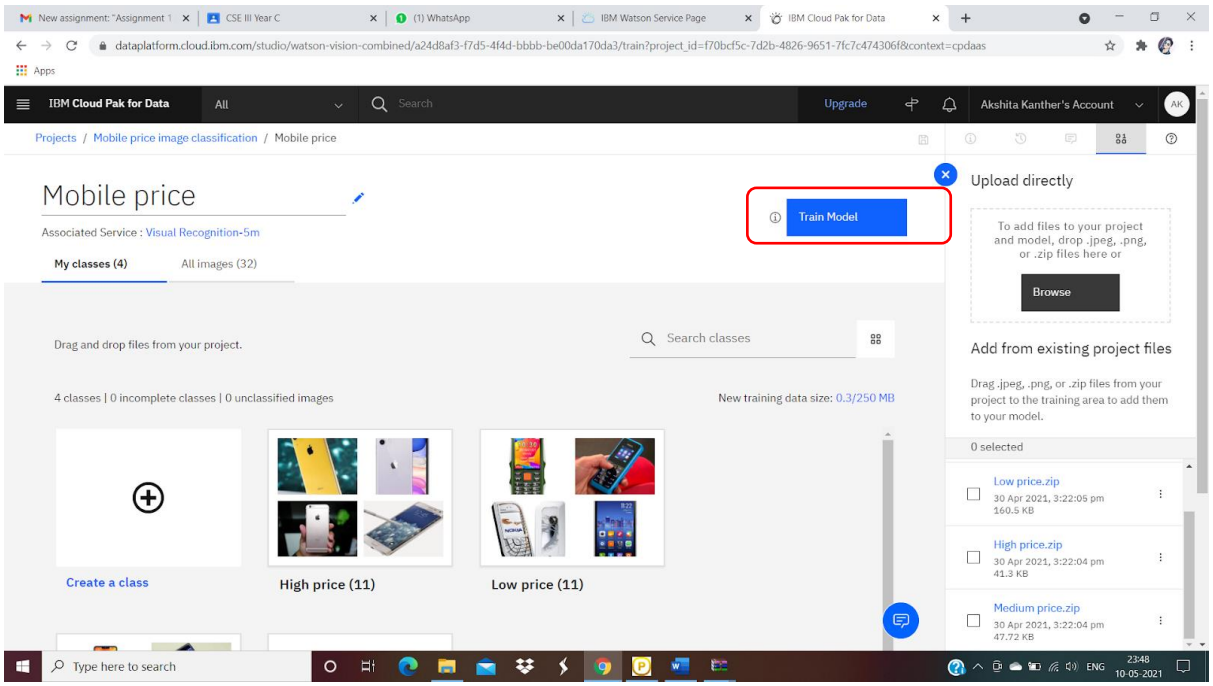


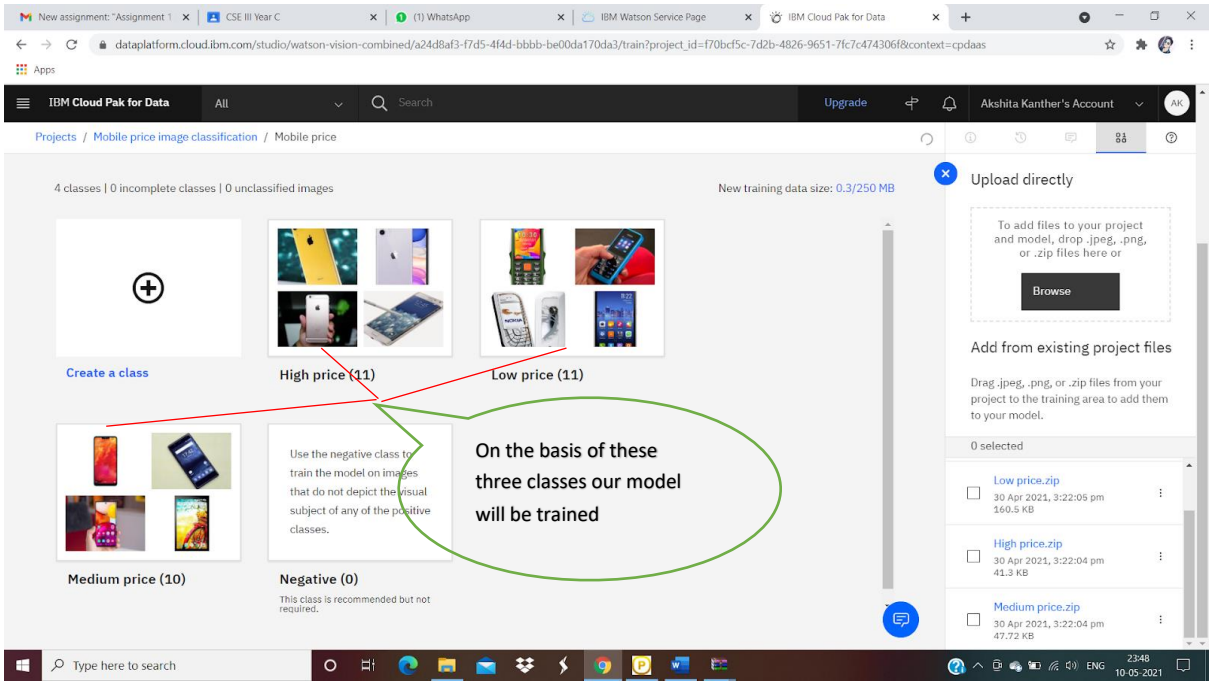


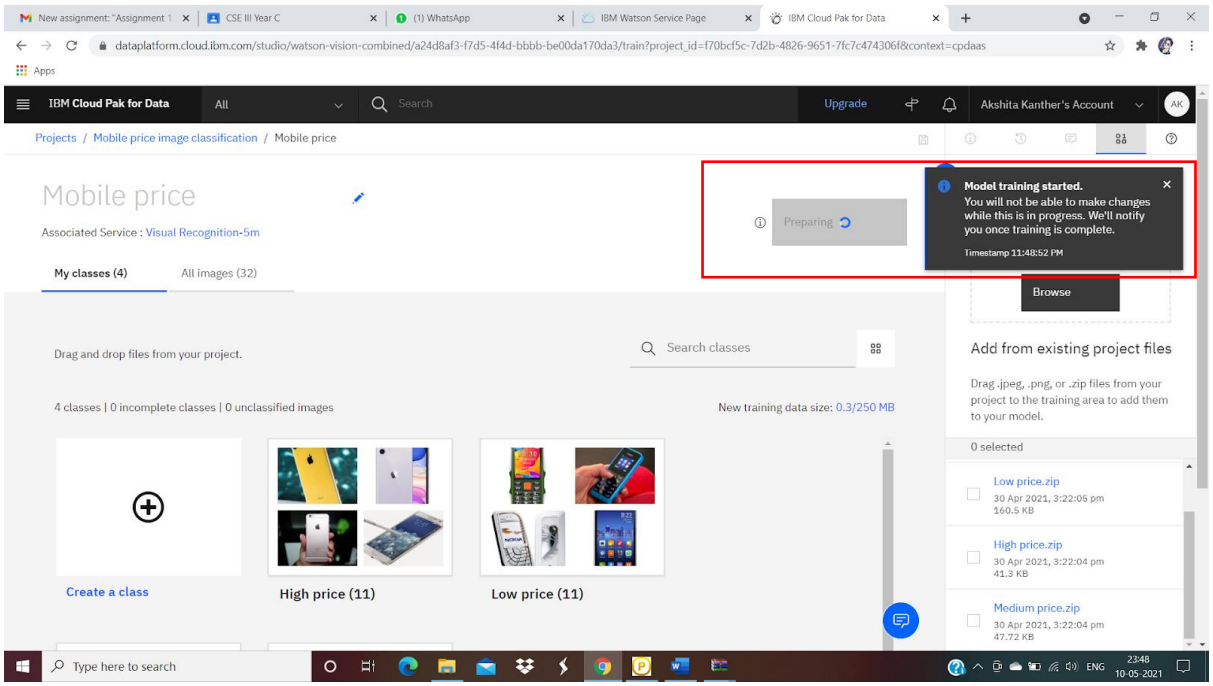
Step-12. Delete the unwanted class which has no images.



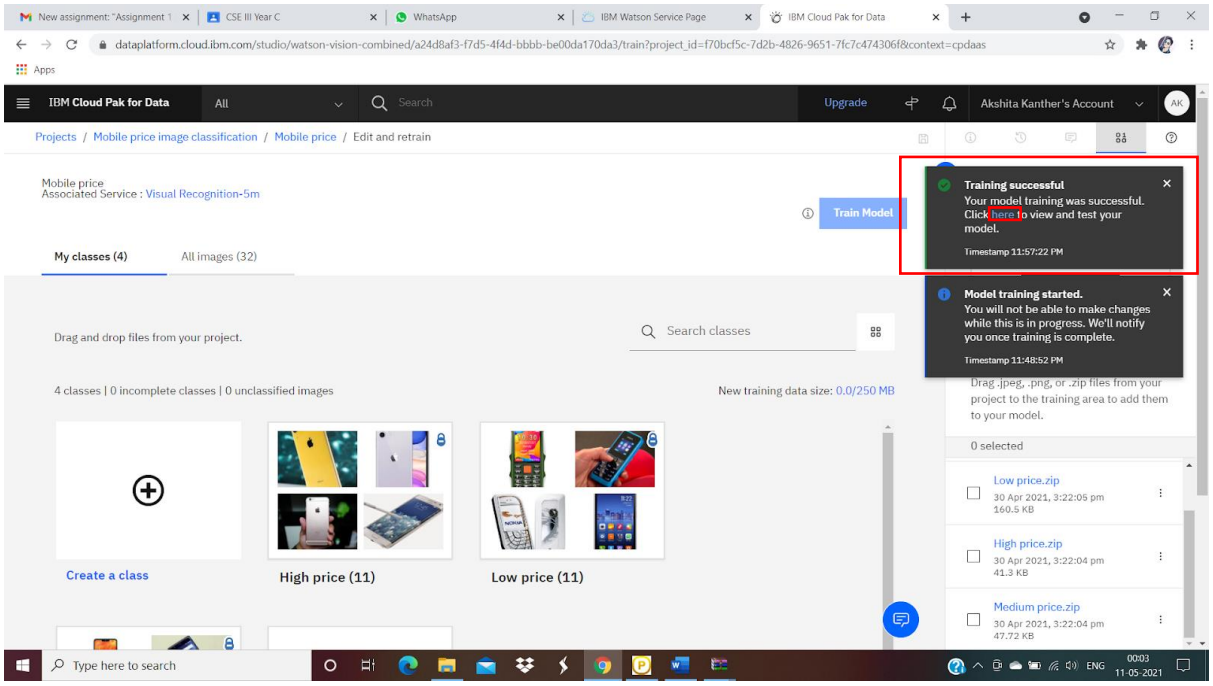
Step-13. Click on Train Model.



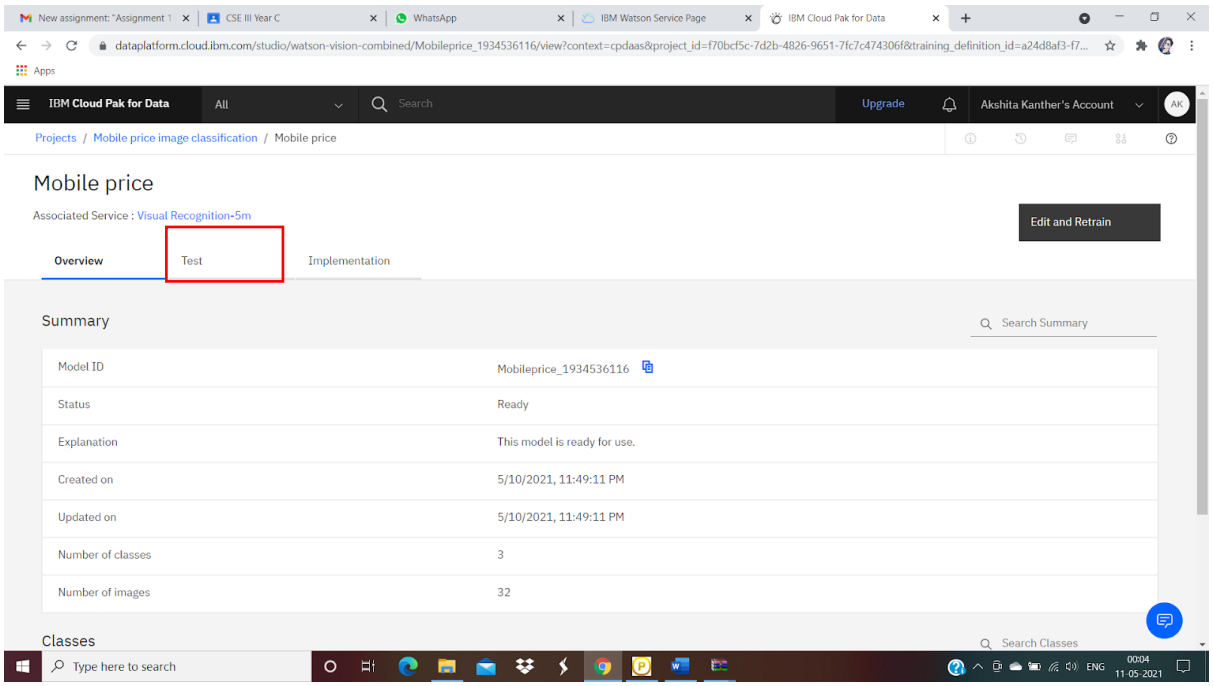




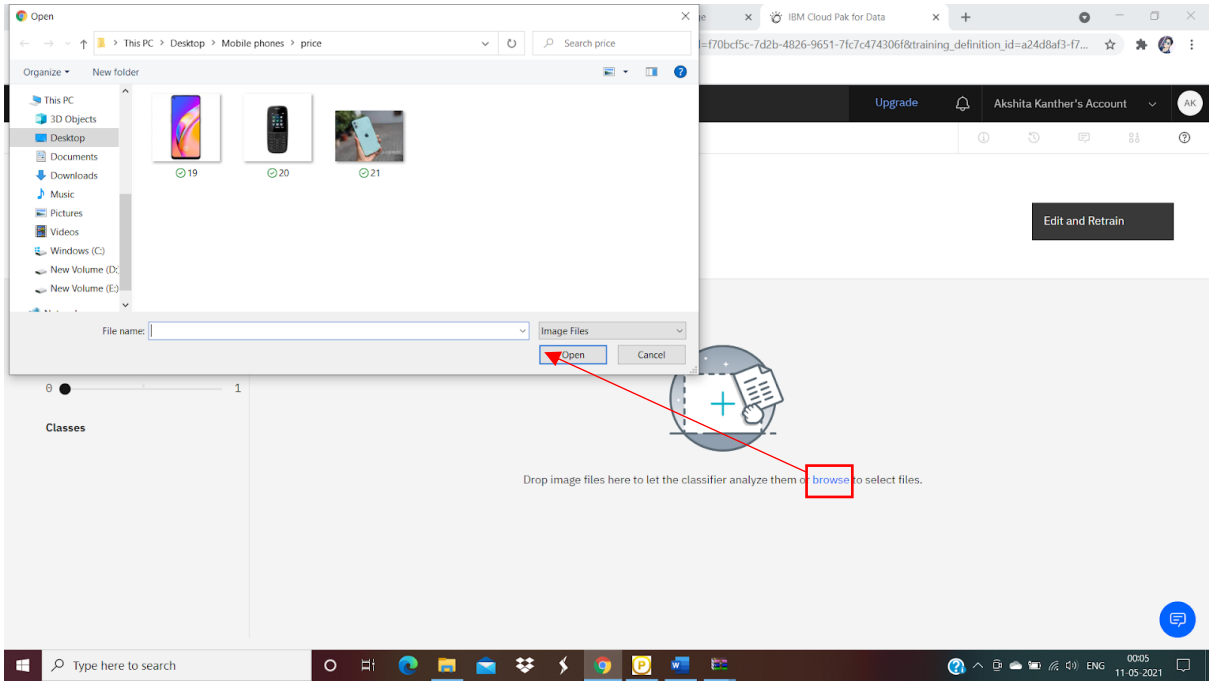
Step-14. After the Training has been completed successfully, a dialog box will pop up containing Training successful message, click on here.

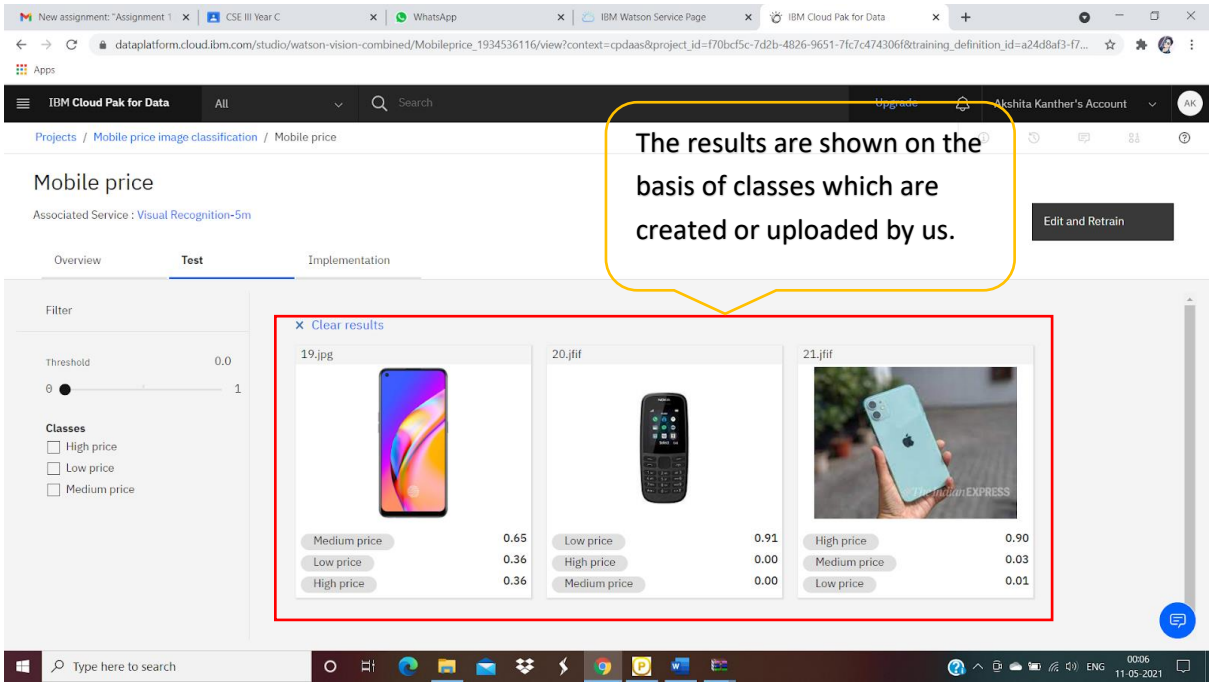


Step-15. Go to Test.



Step-16. Upload the sample test images for different classes: select the images from local folder of your pc and then click on open.





The output is displayed on the basis of different classes we have created. We can observe that our model has detected images correctly.

MACHINE LEARNING MODEL (AutoAI experiment)-

The AutoAI graphical tool in Watson Studio automatically analyses our data and generates candidate model pipelines customized for our predictive modelling problem. These model pipelines are created iteratively as AutoAI analyses your dataset and discovers data transformations, algorithms, and parameter settings that work best for our problem setting. Results are displayed on a leader board, showing the automatically generated model pipelines ranked according to our problem optimization objective.

- **Required service**

  Watson Machine Learning service

- **Data format**

  Tabular: CSV files, with comma (,) delimiter

- **Data size**

  Less than 1 GB

- **AutoAI process**

  Using AutoAI, we can build and deploy a machine learning model with sophisticated training features and no coding. The tool does most of the work for us.

AutoAI automatically runs the following tasks to build and evaluate candidate model pipelines:

- **Data pre-processing**

Most data sets contain different data formats and missing values, but standard machine learning algorithms work with numbers and no missing values. AutoAI applies various algorithms, or estimators, to analyse, clean, and prepare your raw data for machine learning. It automatically detects and categorizes features based on data type, such as categorical or numerical. Depending on the categorization, it uses hyper-parameter optimization to determine the best combination of strategies for missing value imputation, feature encoding, and feature scaling for your data.

- **Automated model selection**

The next step is automated model selection that matches your data. AutoAI uses a novel approach that enables testing and ranking candidate algorithms against small subsets of the data, gradually increasing the size of the subset for the most promising algorithms to arrive at the best match. This approach saves time without sacrificing performance. It enables ranking a large number of candidate algorithms and selecting the best match for the data.

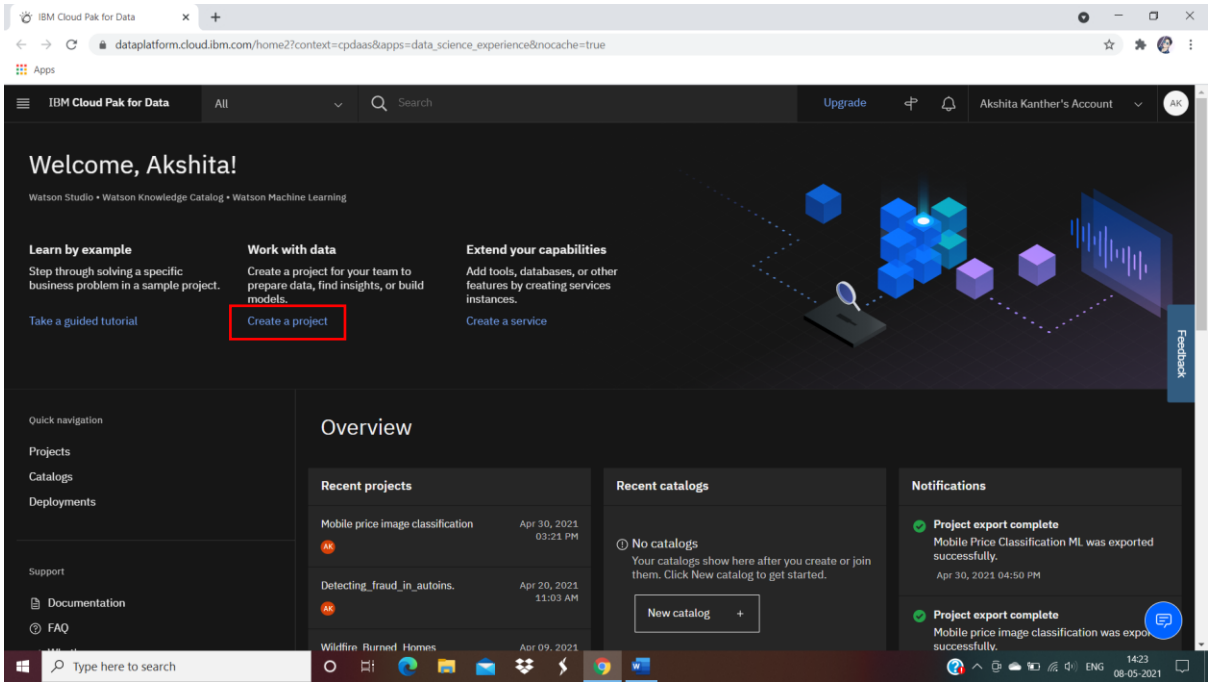
- **Automated feature engineering**

Feature engineering attempts to transform the raw data into the combination of features that best represents the problem to achieve the most accurate prediction. AutoAI uses a unique approach that explores various feature construction choices in a structured, non-exhaustive manner, while progressively maximizing model accuracy using reinforcement learning. This results in an optimized sequence of transformations for the data that best match the algorithms of the model selection step.

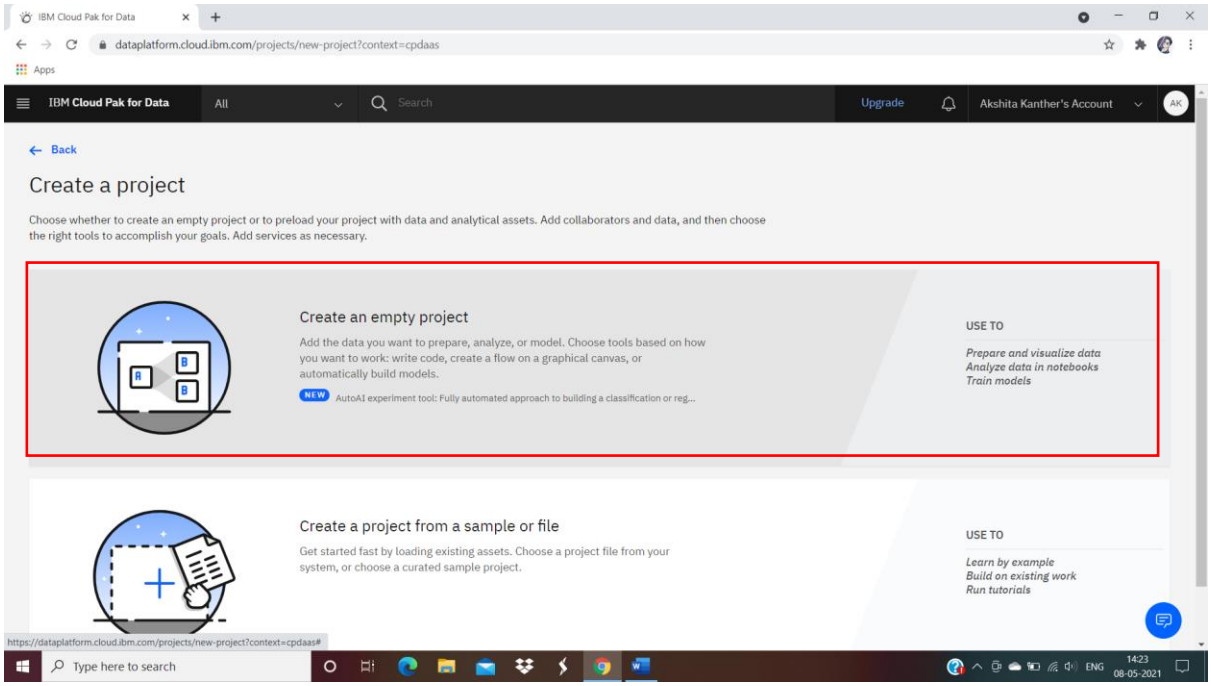
- **Hyper parameter optimization**

Finally, a hyper-parameter optimization step refines the best performing model pipelines. AutoAI uses a novel hyper-parameter optimization algorithm optimized for costly function evaluations such as model training and scoring that are typical in machine learning. This approach enables fast convergence to a good solution despite long evaluation times of each iteration.

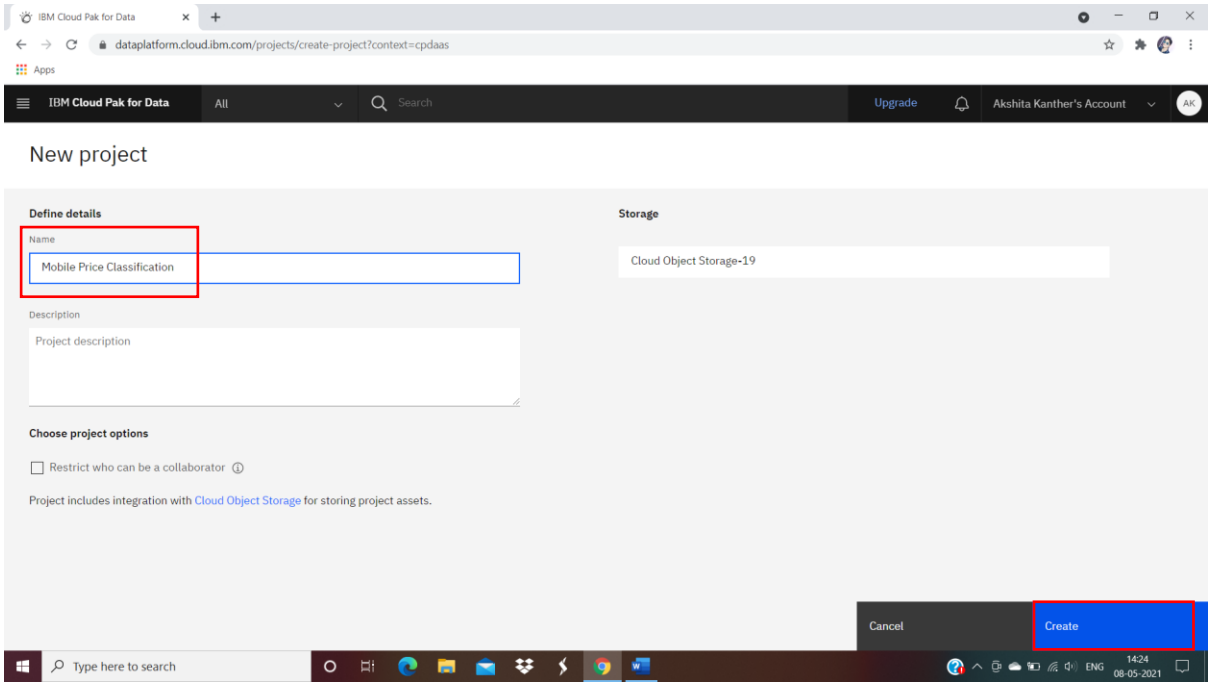
Step-1. Login to IBM cloud platform and **launch IBM Watson** service from resources (if IBM Watson service instance is not created then we will first create it and then launch it). Click on **Create a project**.



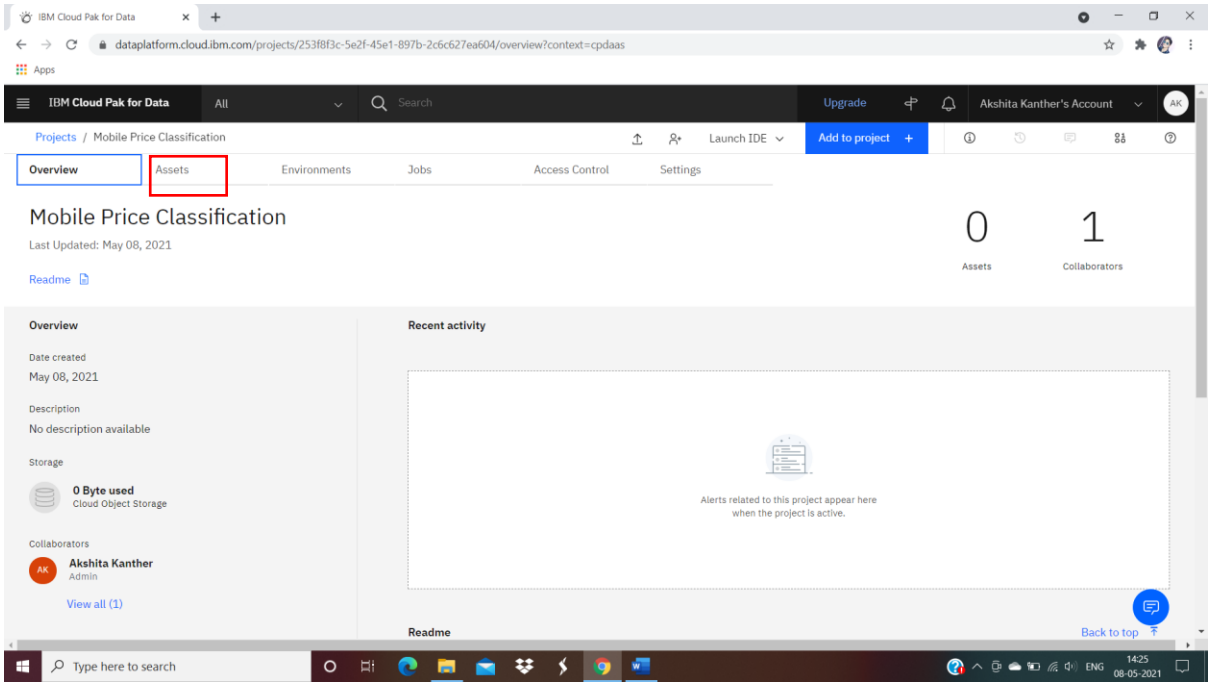
Step-2. Click on create an empty project.



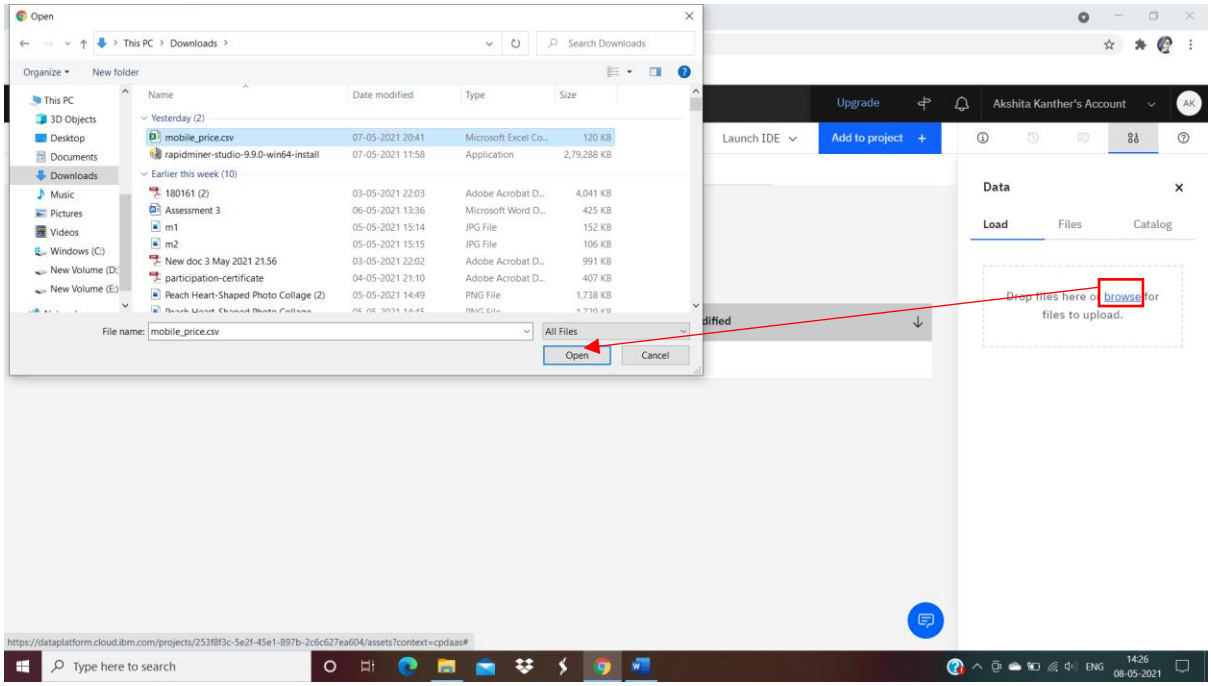
Step-3. Enter your project name. Also ensure that an instance of Cloud Object Storage is created. Enter project description (optional). Then click on Create.



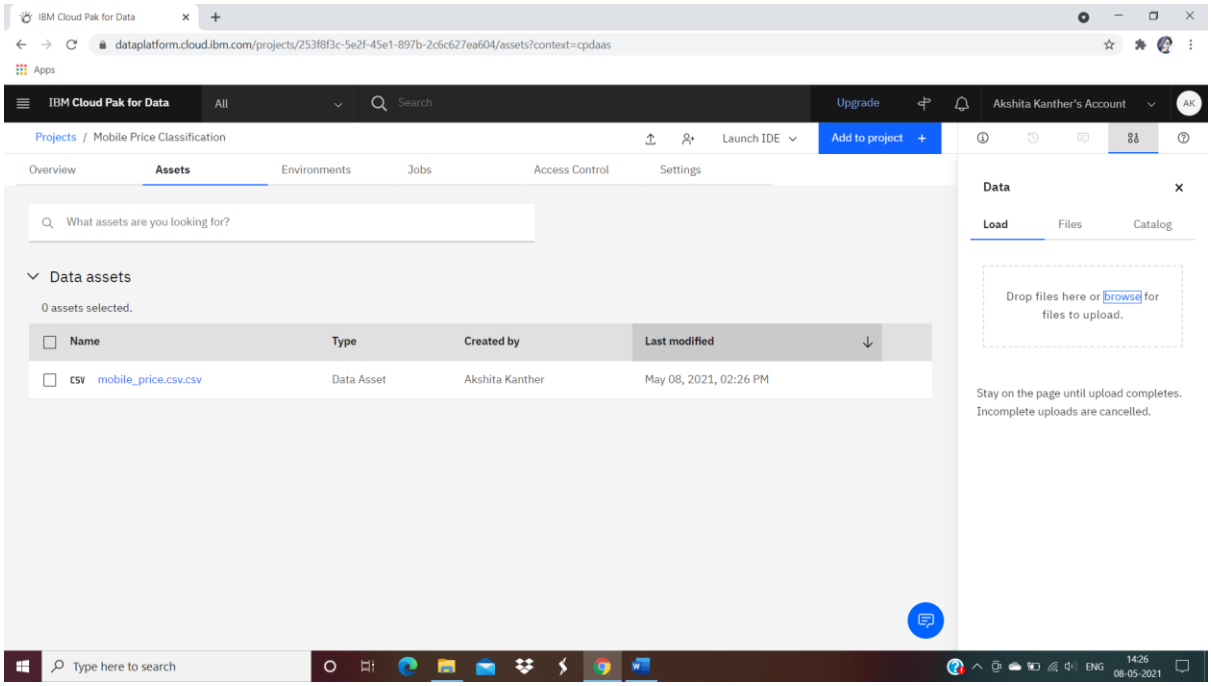
Step-4. Click on Assets.



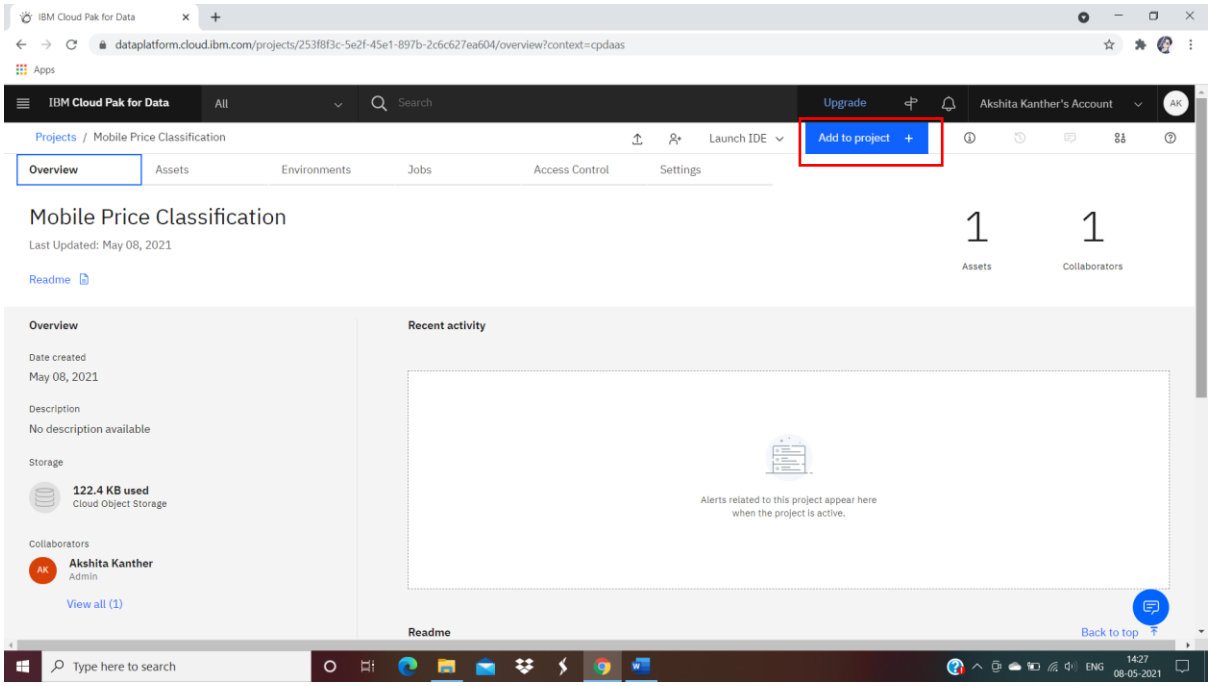
Step-5. Click on browse under load in the Data pane on the left side of the page. And upload your dataset in CSV format.



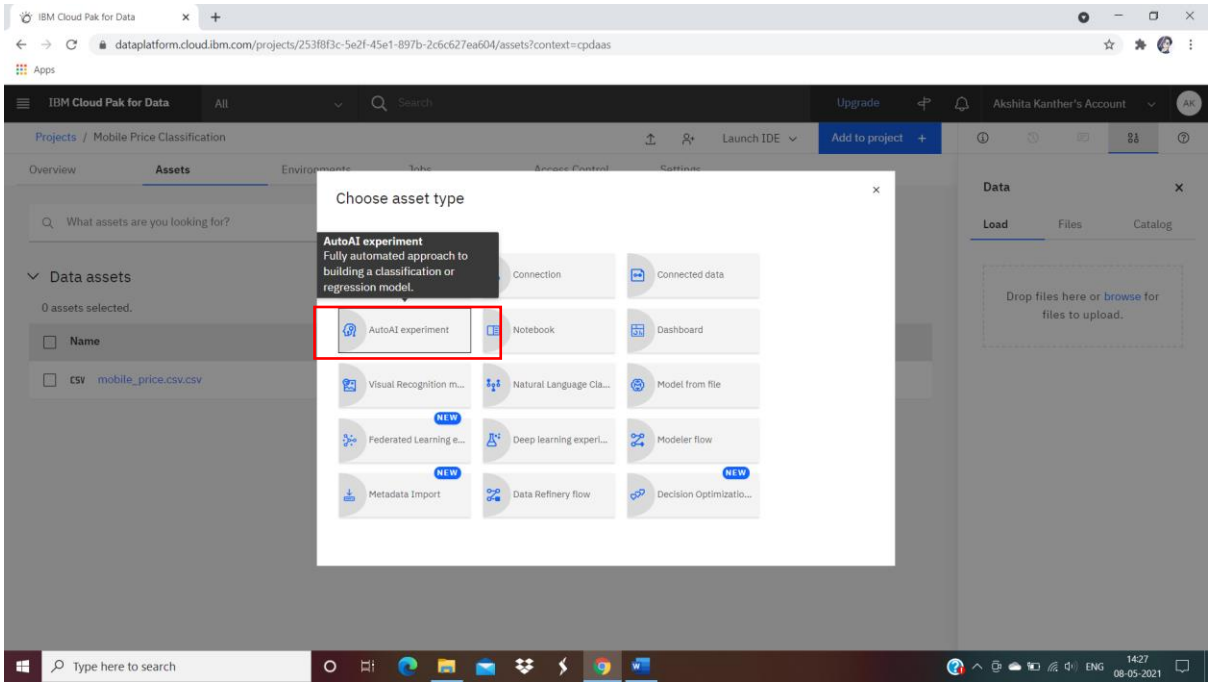
Step-6. Click on Overview .



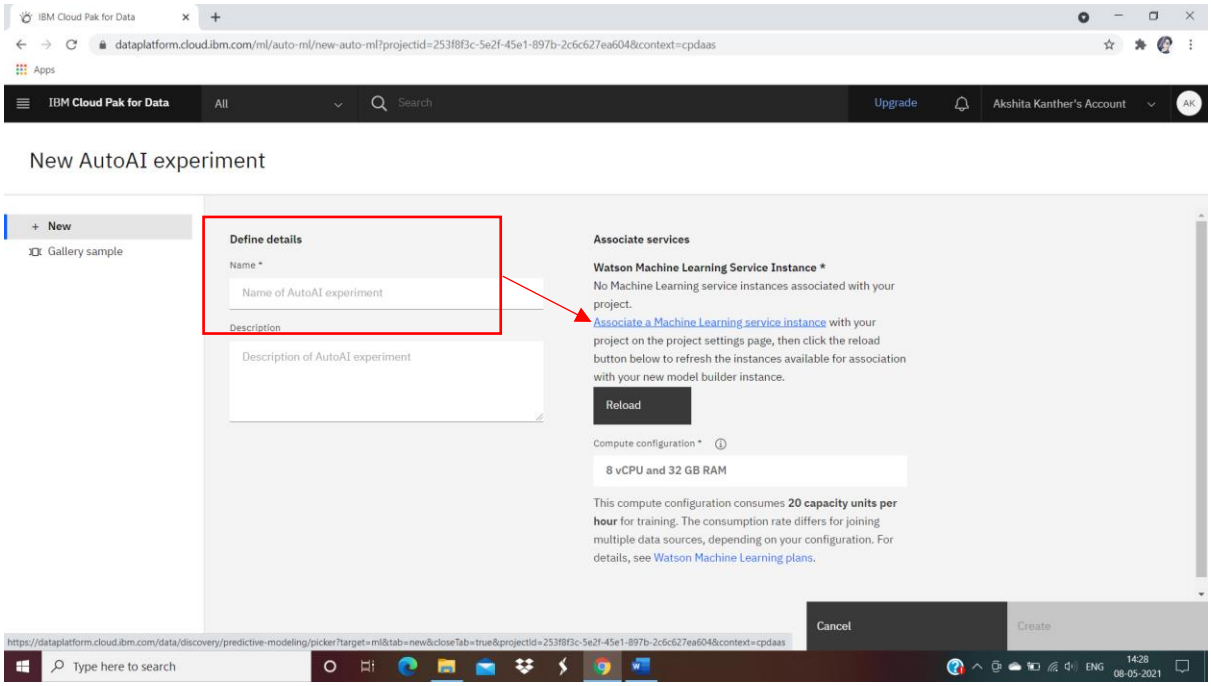
Step-7. Now we can see that 1 Asset has been uploaded and the no. above Assets has been updated. Now click on Add to project which is on top right side of the page.



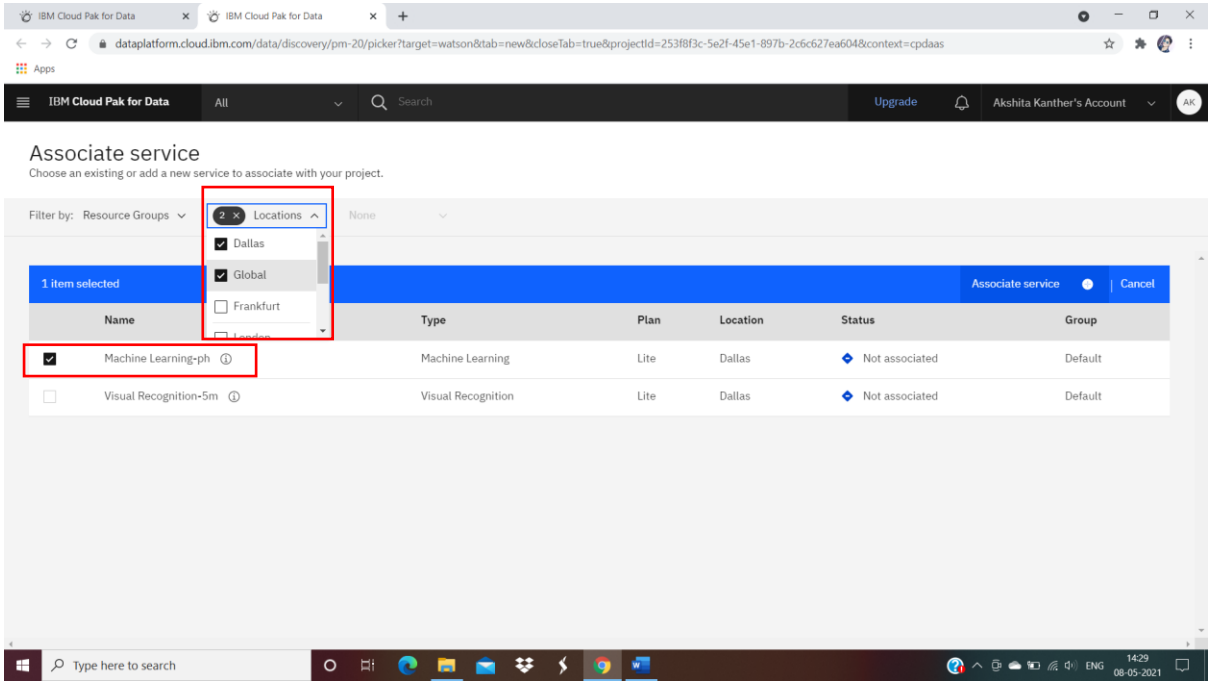
Step-8. Click on AutoAI experiment.



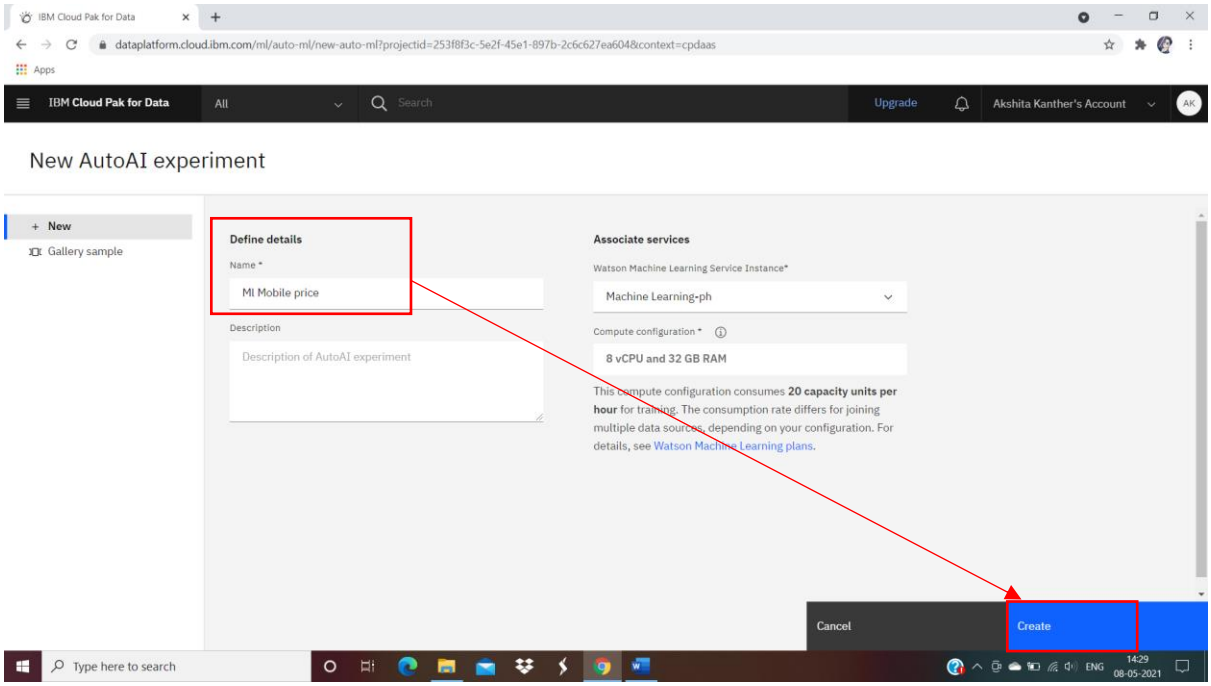
Step-9. Enter the name of the New AutoAI experiment, and click on Associate a Machine Learning service instance.

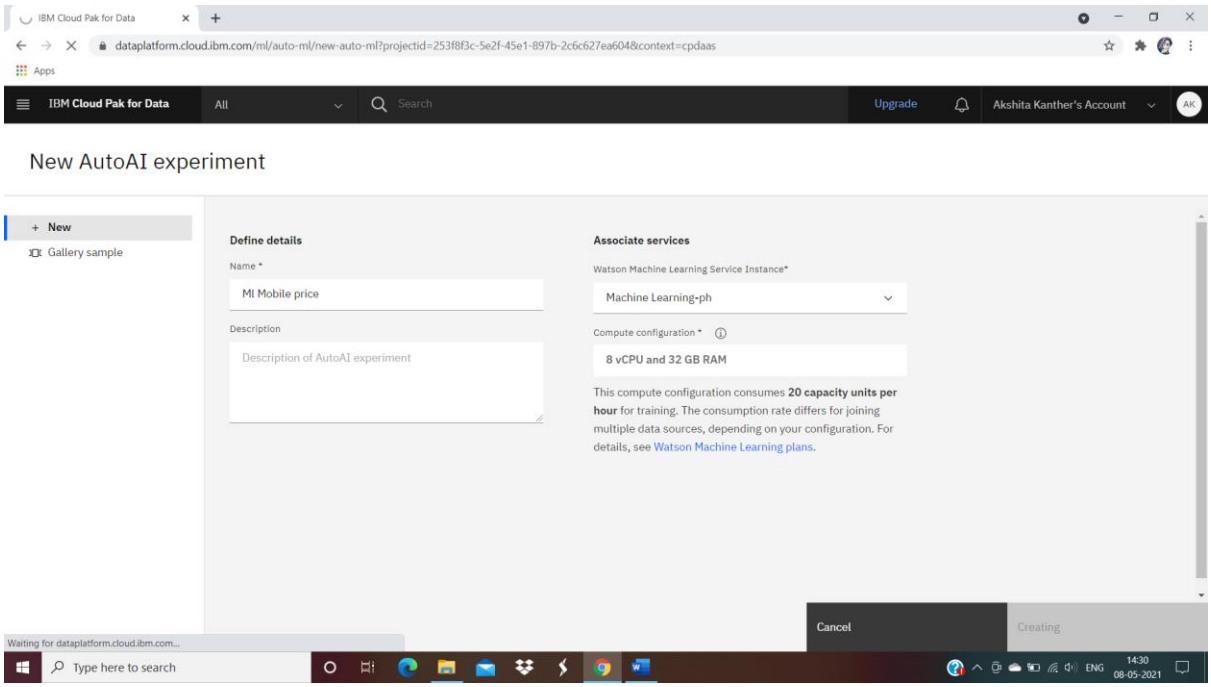


Step-10. Select the Machine learning service instance, in Locations select Dallas and Global, then click on Associate service and then Reload the Machine learning service instance.

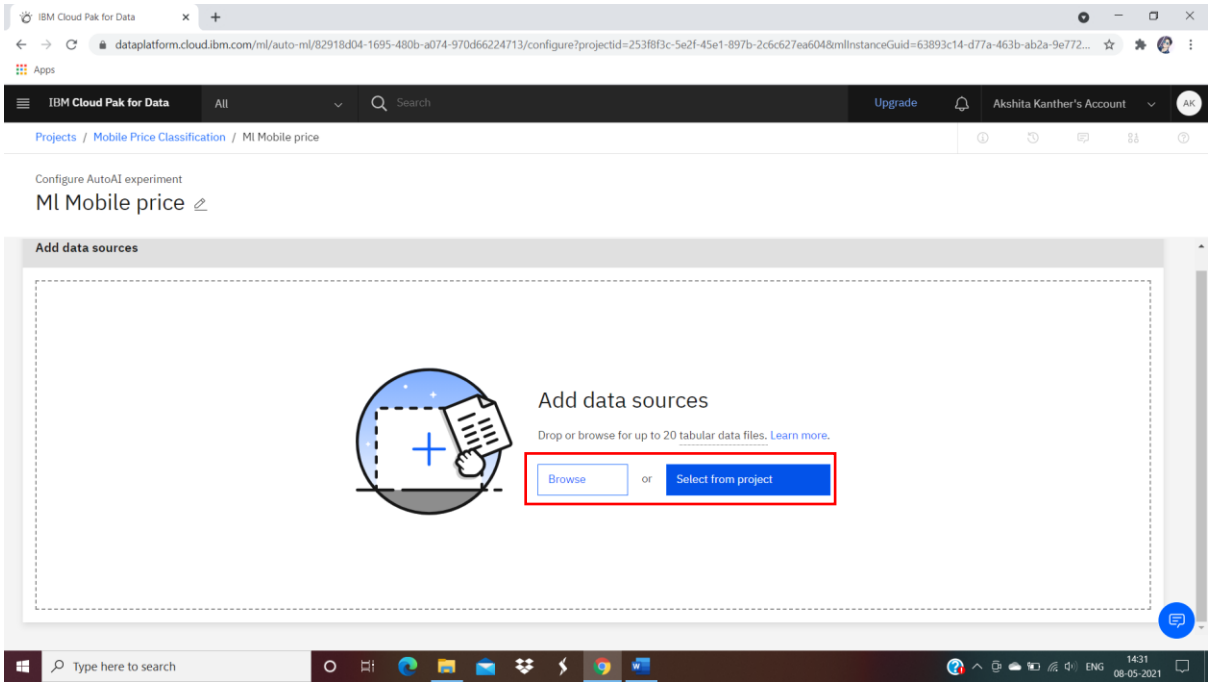


Step-11. Click on Create.

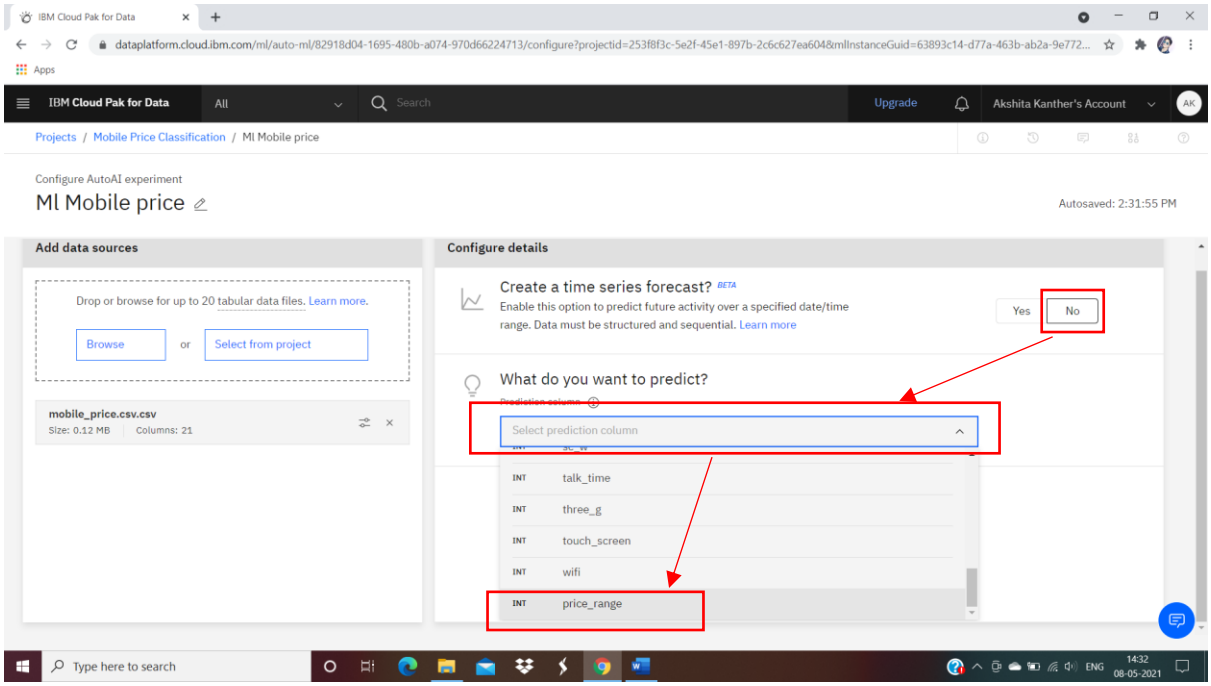




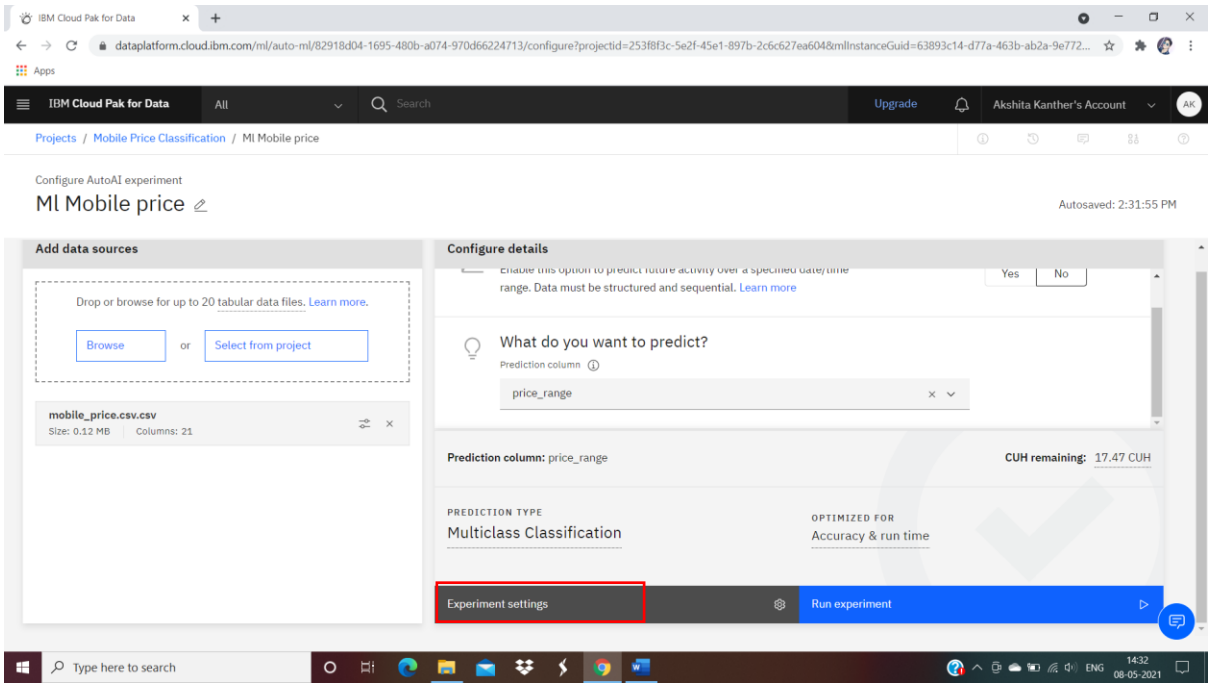
Step-12. Click on Browse and upload the dataset.



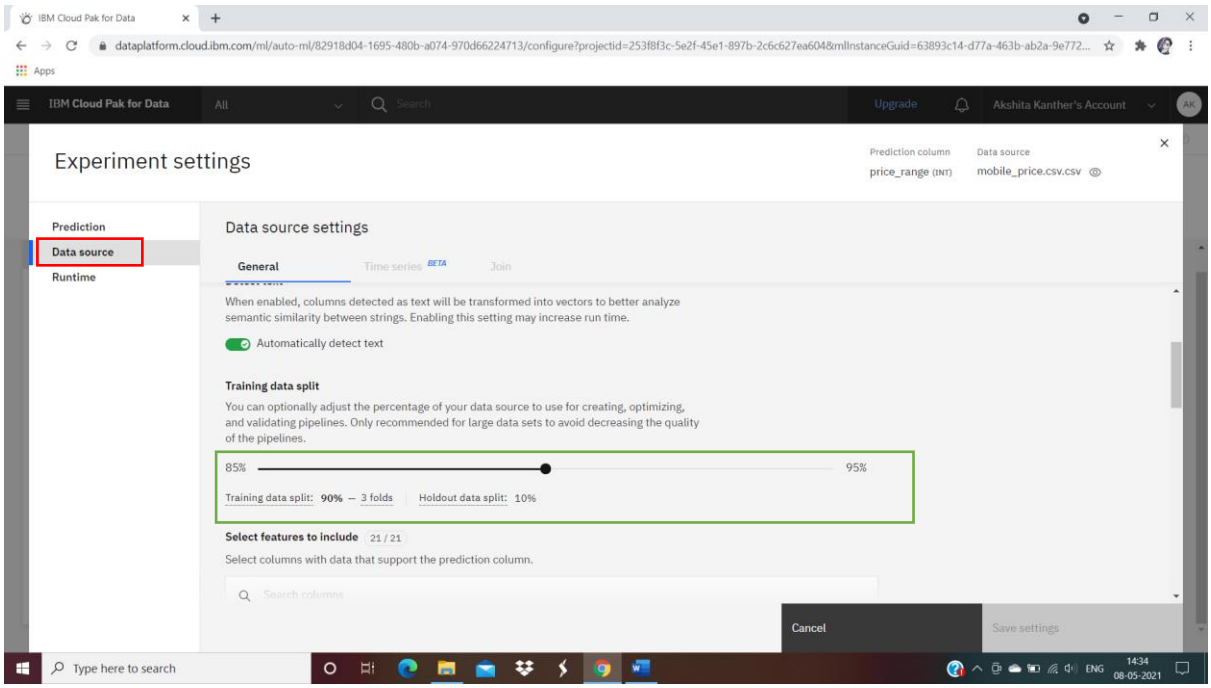
Step-13. After dataset has been uploaded, select No in Create a time series forecast? Then click on dropdown under What do you want to predict? Select the column that you want to predict.



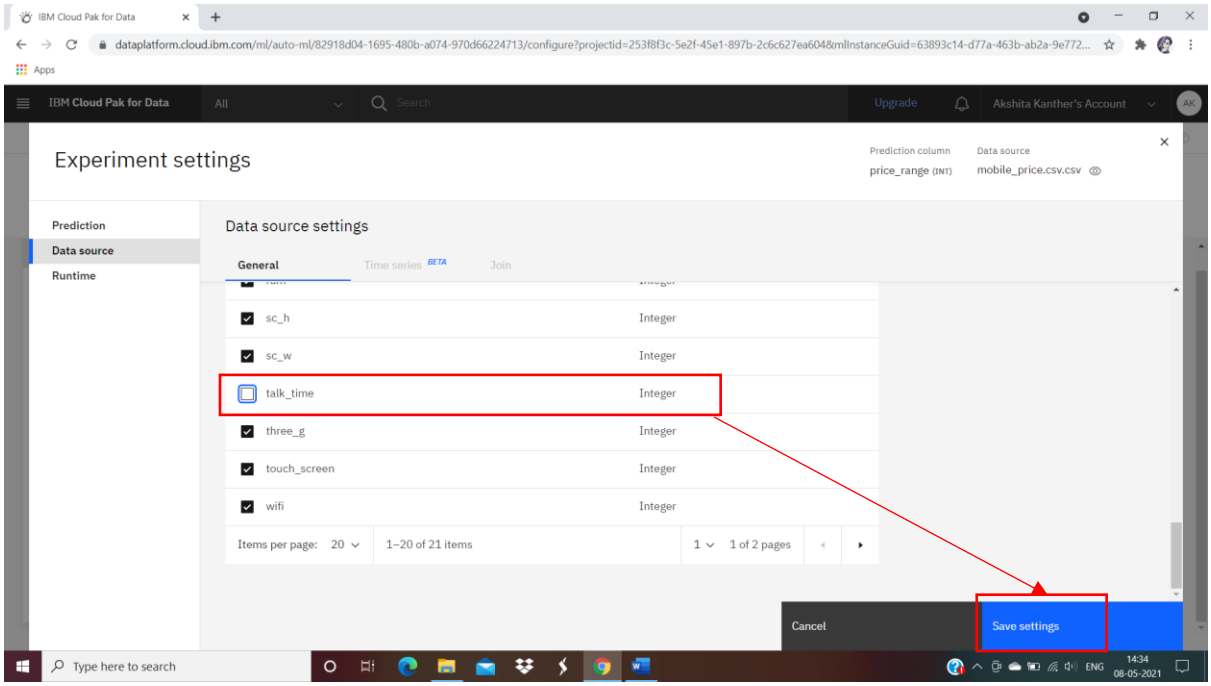
Step-14. It will automatically detect the PREDICTION TYPE, then click on Experiment settings at the bottom.



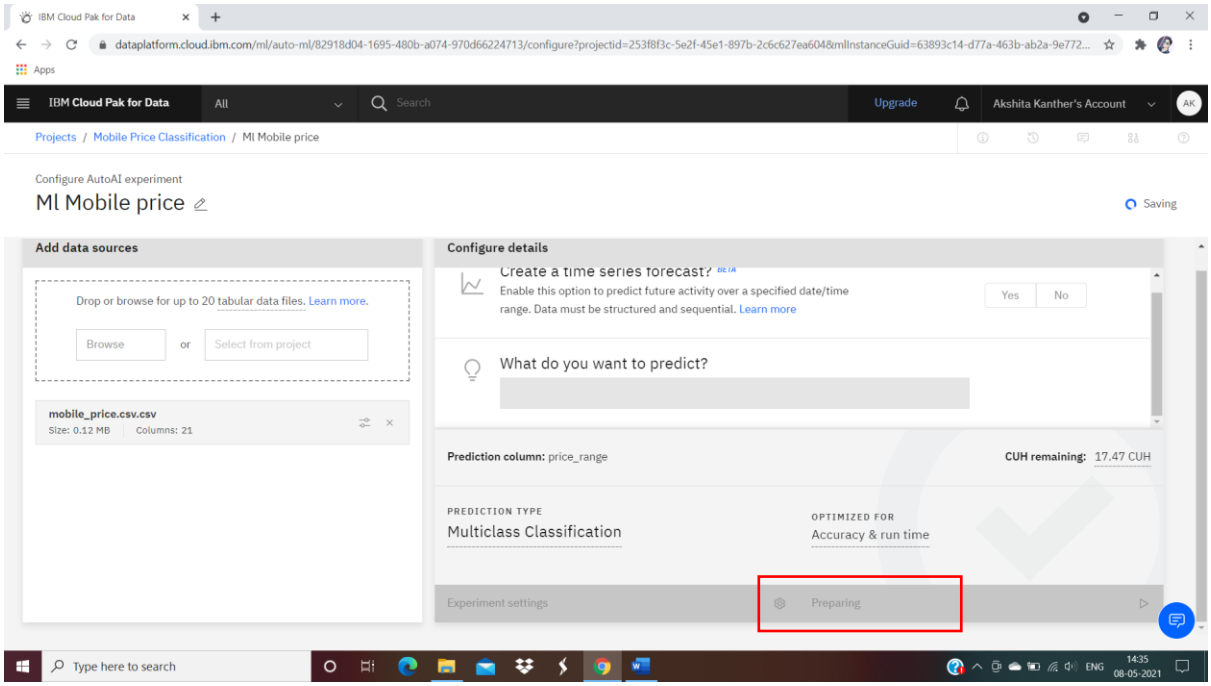
Step-15. Click on Data Source on the left side. By default training and testing data will be split in the ratio of 90% to 10%. This ratio is preferred, we can change according to our requirement by adjusting the scale.

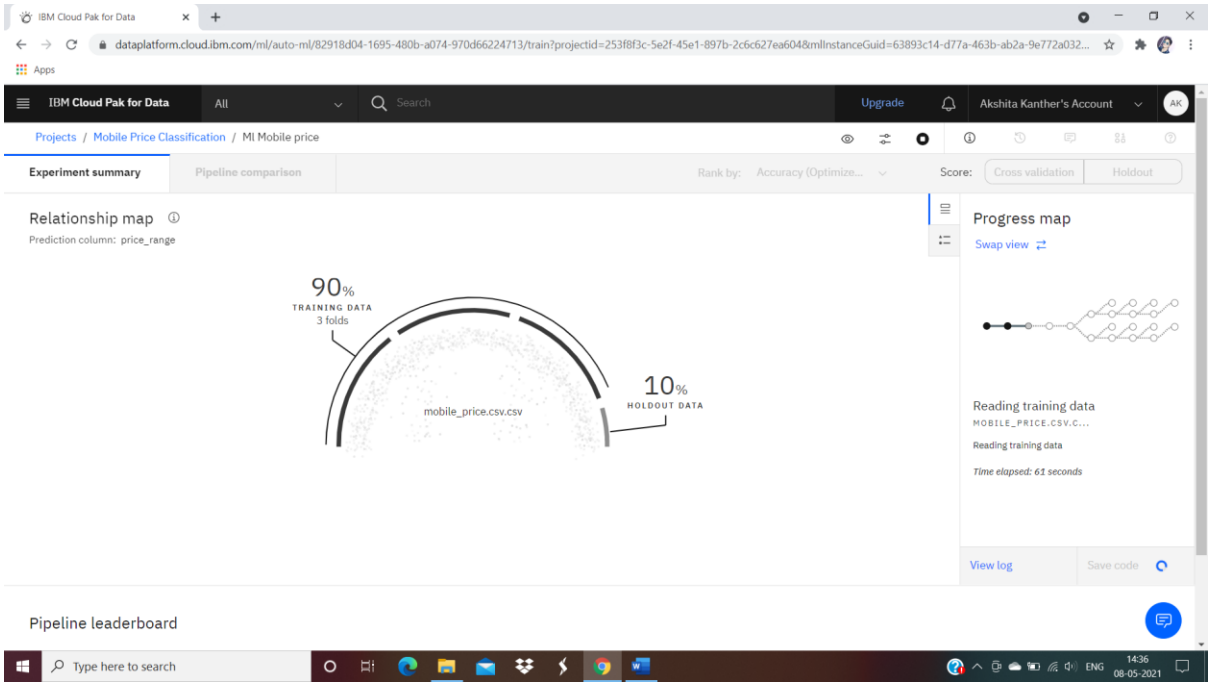


Step-16. Unselect the column names which aren't required for making the model. And then click on Save settings.

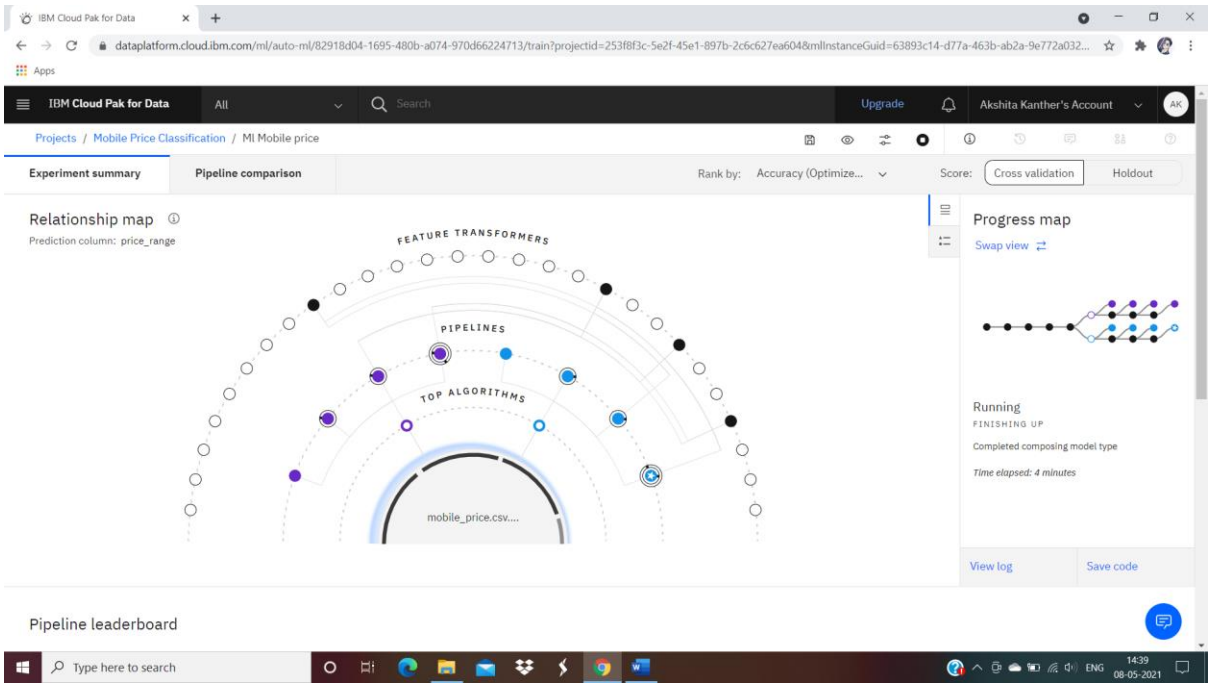


Step-17. Click on Run experiment at the bottom of the page.

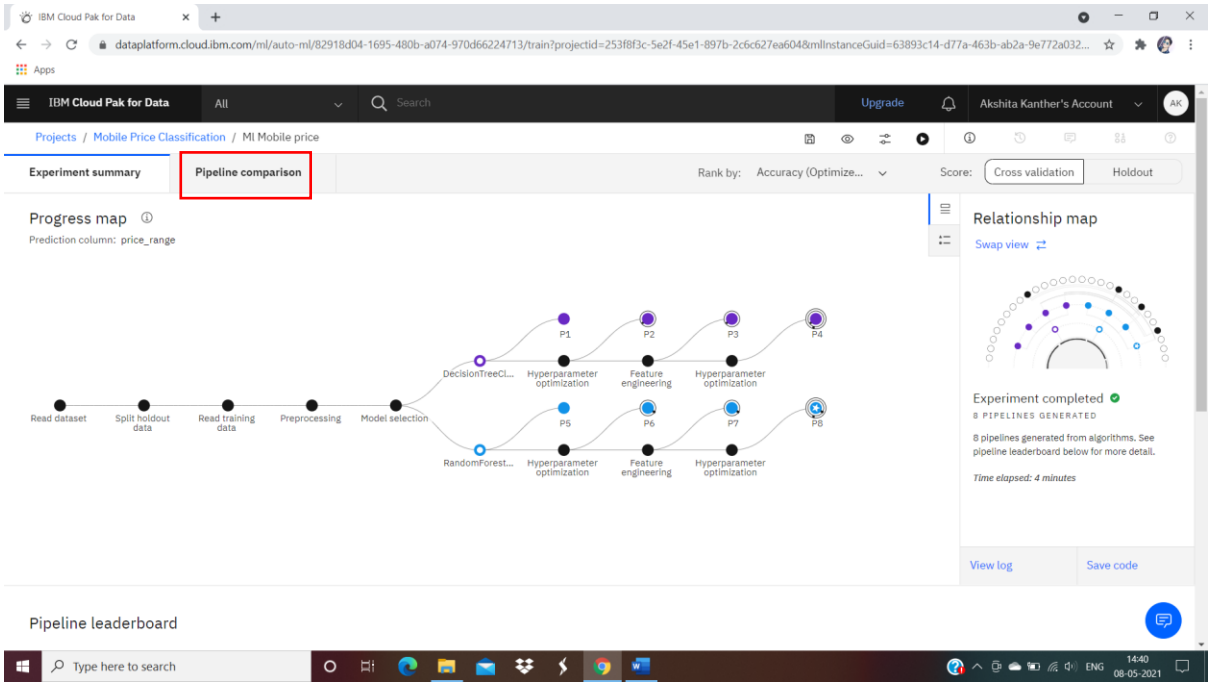


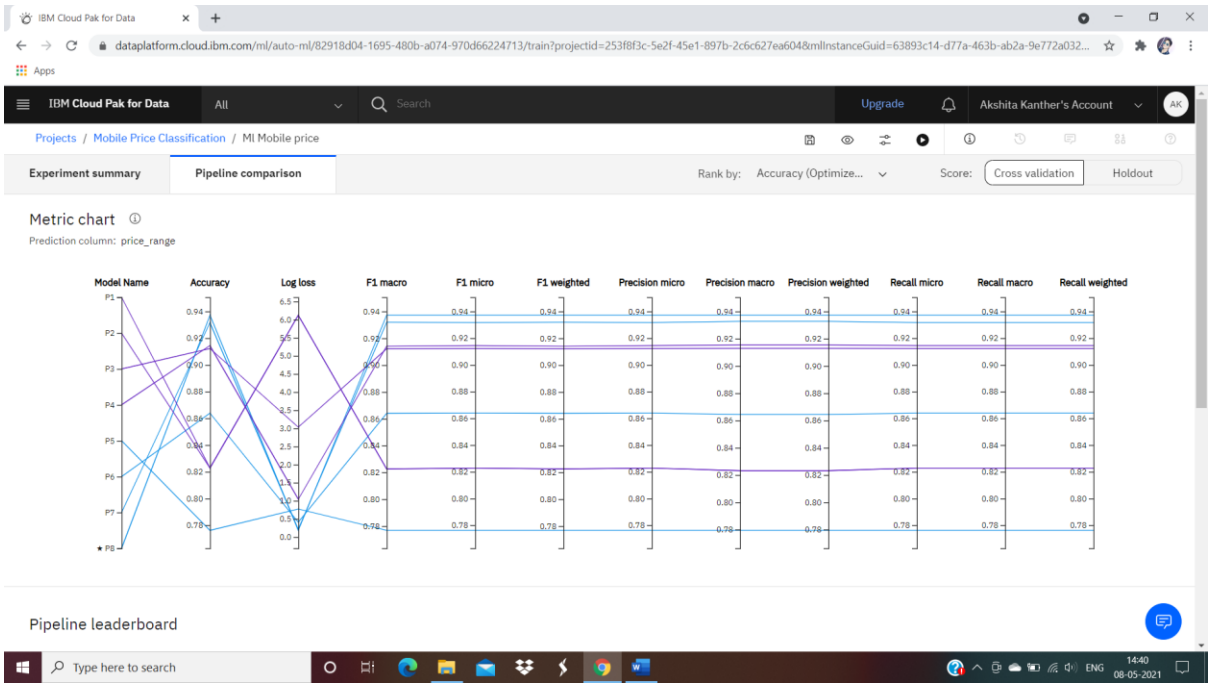


Step-18. Pipelines have been generated for the model. Pipeline with * indicates the best pipeline. Click on Swap view under Progress map on the right side of the screen.

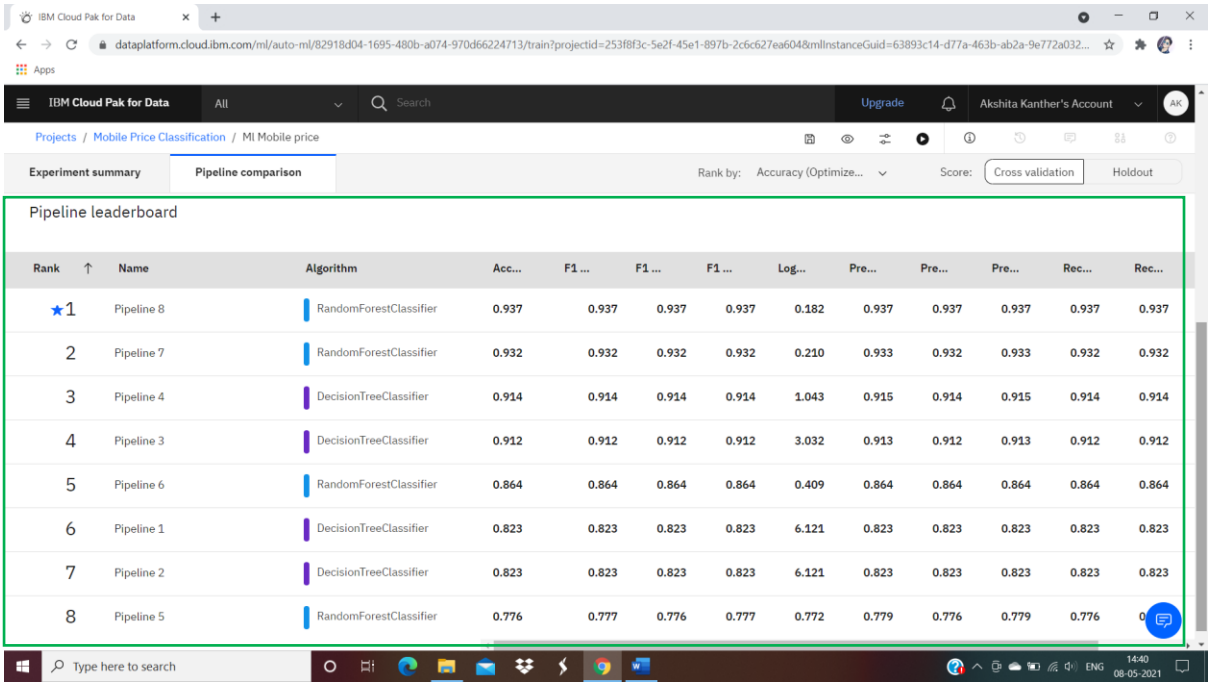


Step-19. Click on Pipeline comparison on the top.

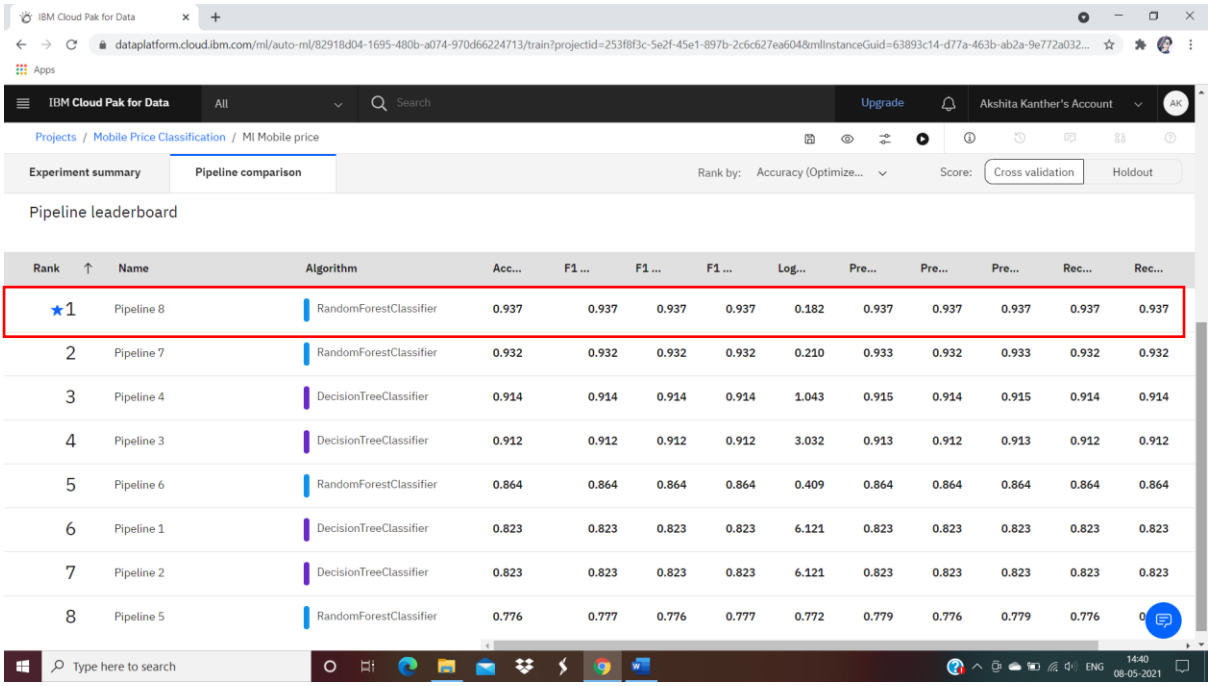


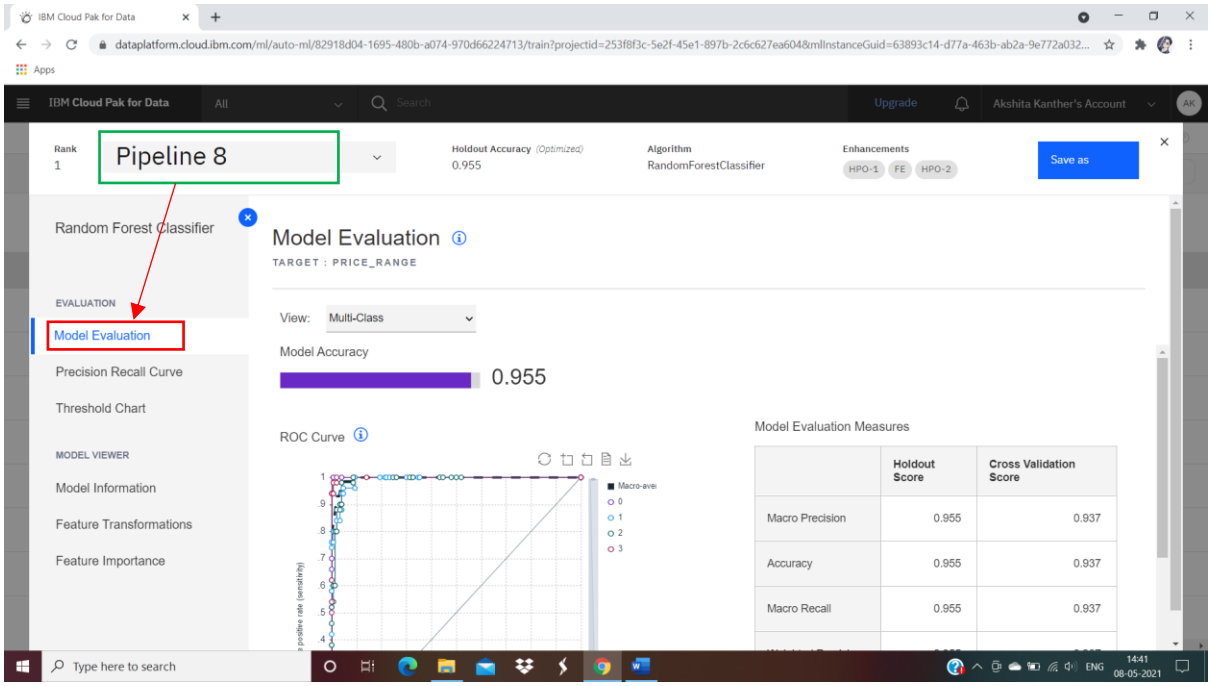


Step-20. Scroll down and view Pipeline leaderboard.

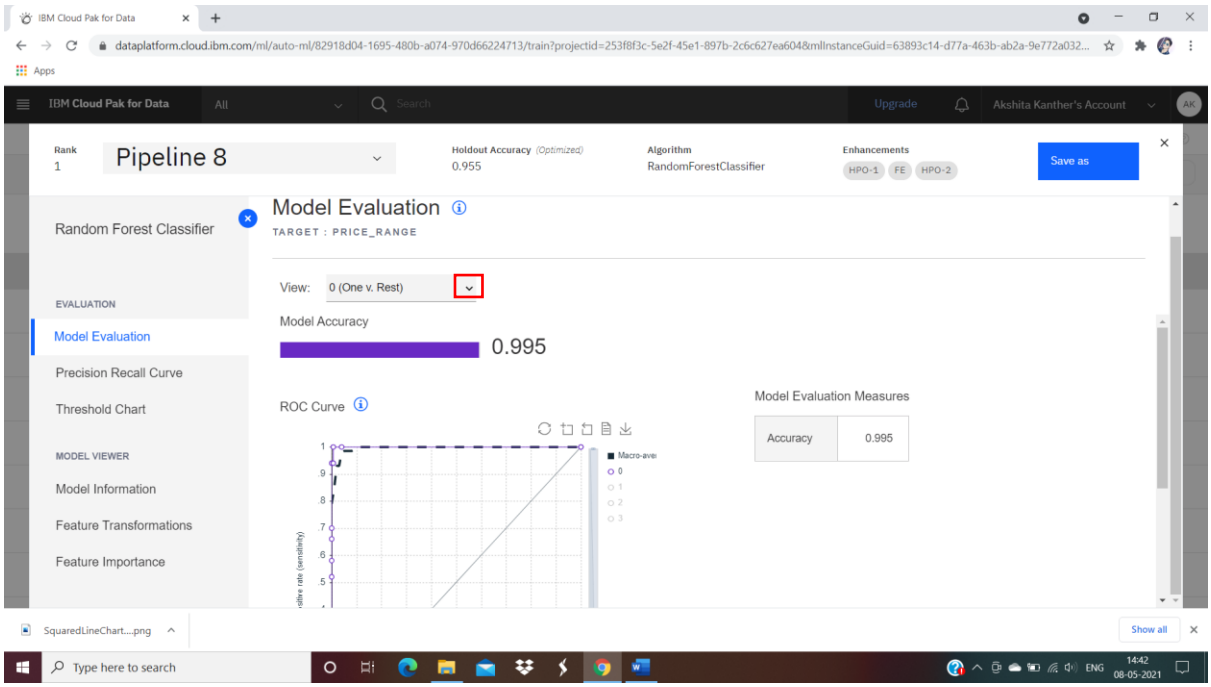


Step-21. View pipeline with 1st rank, it is the best pipeline with a * and best results. Click on it. Then click on Model Evaluation which is on the left side of the screen. Model accuracy, ROC curve and Model Evaluation measures are displayed on the screen.

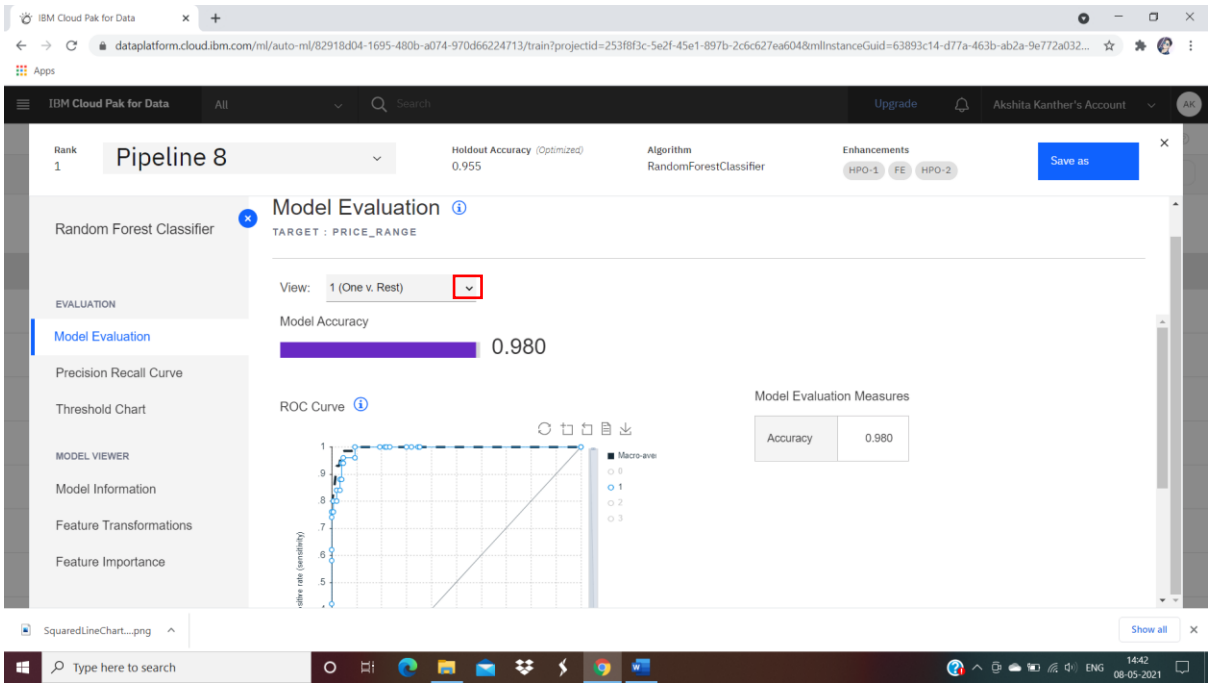




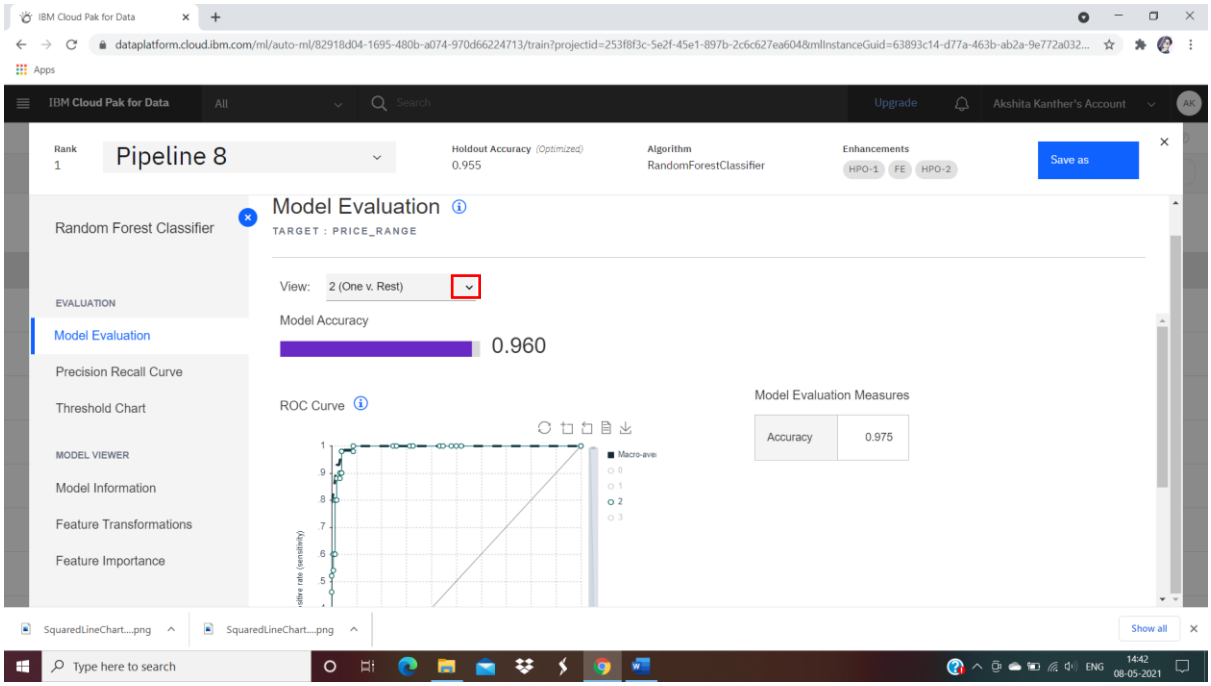
Step-22. Click on the dropdown which is present in front of the view and select 0(One v. Rest) .View the output for one class of the predicted column.



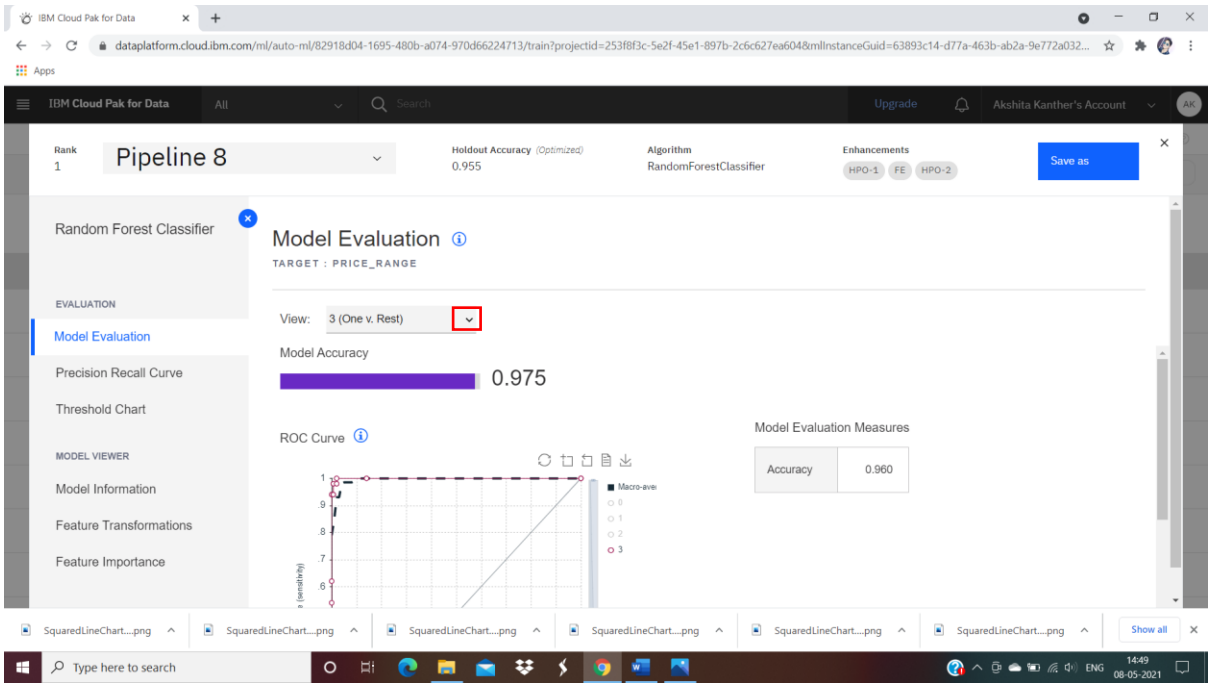
Step-23. Click on the dropdown which is present in front of the view and select 1(One v. Rest) .View the output for one class of the predicted column.

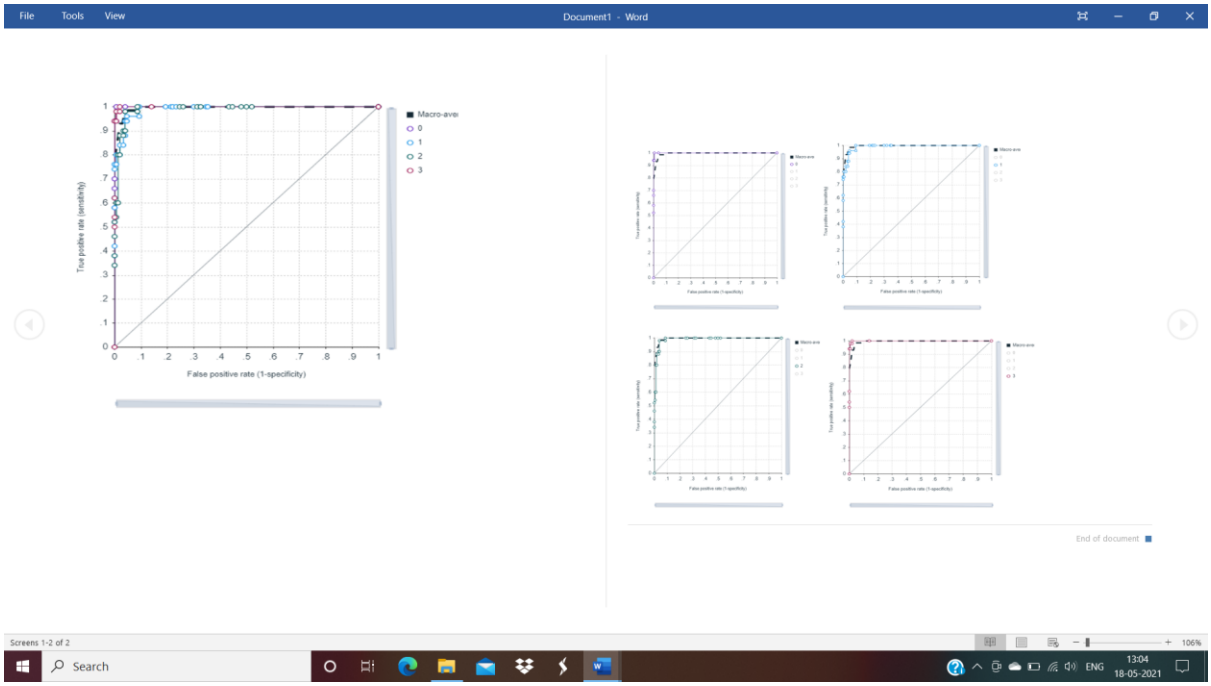


Step-24. Click on the dropdown which is present in front of the view and select 2(One v. Rest) .View the output for one class of the predicted column.

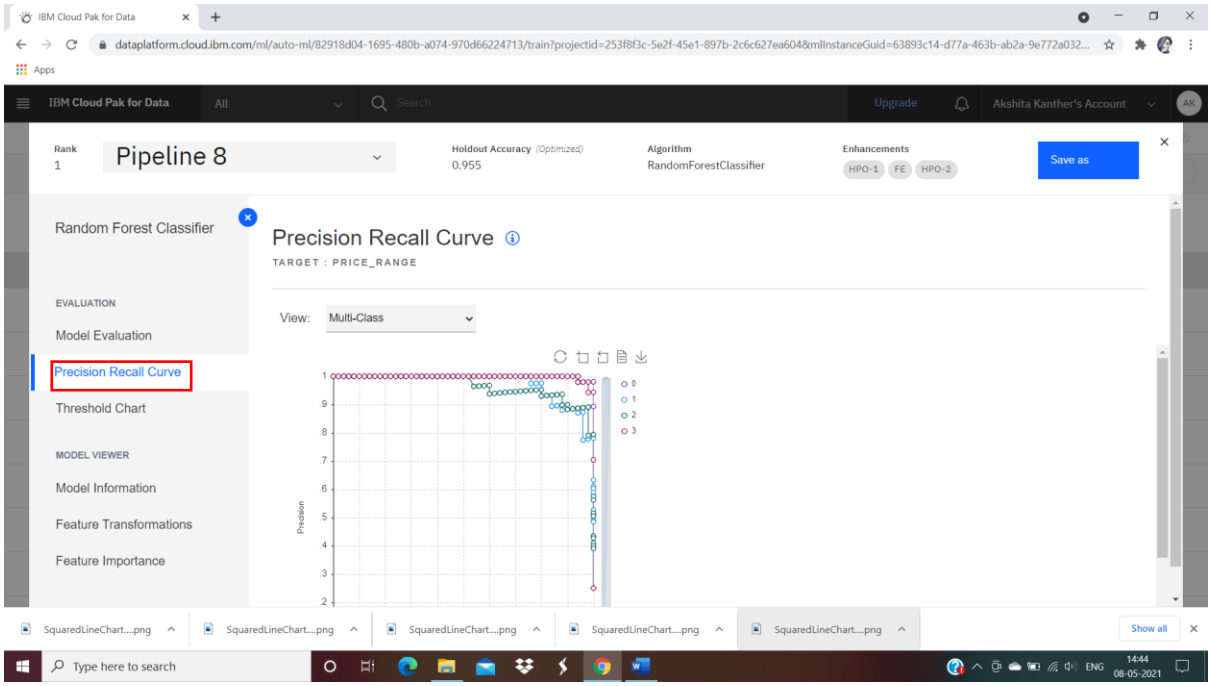


Step-25. Click on the dropdown which is present in front of the view and select 3(One v. Rest) .View the output for one class of the predicted column.

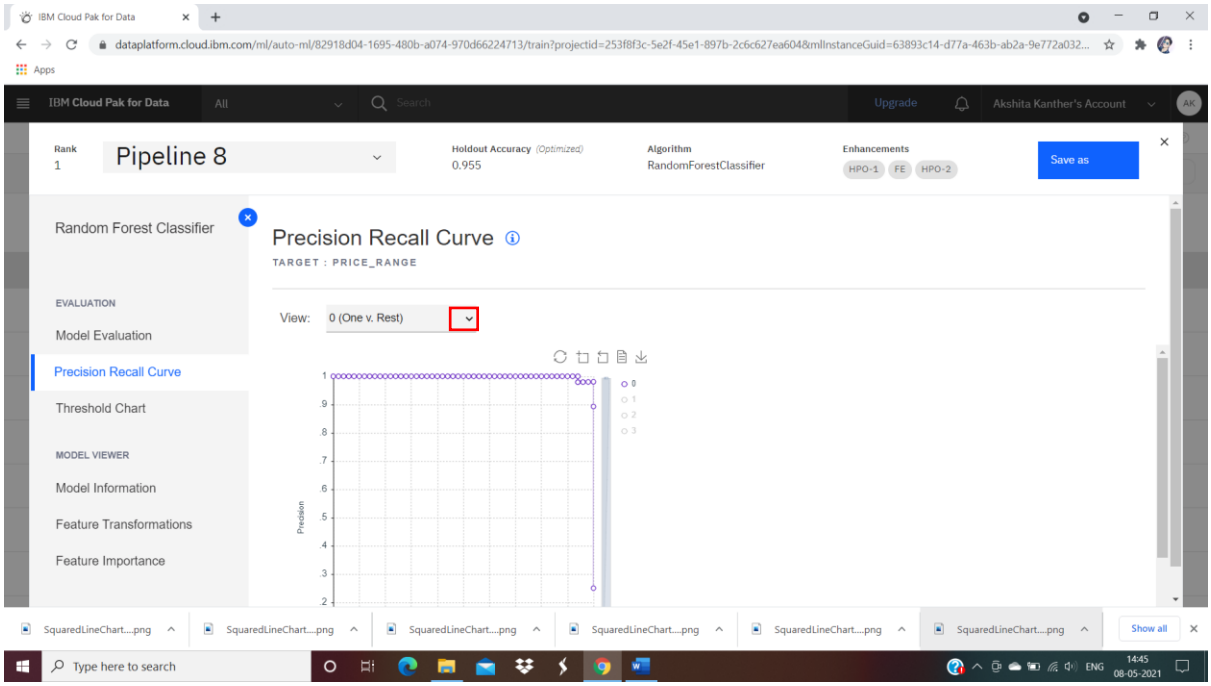




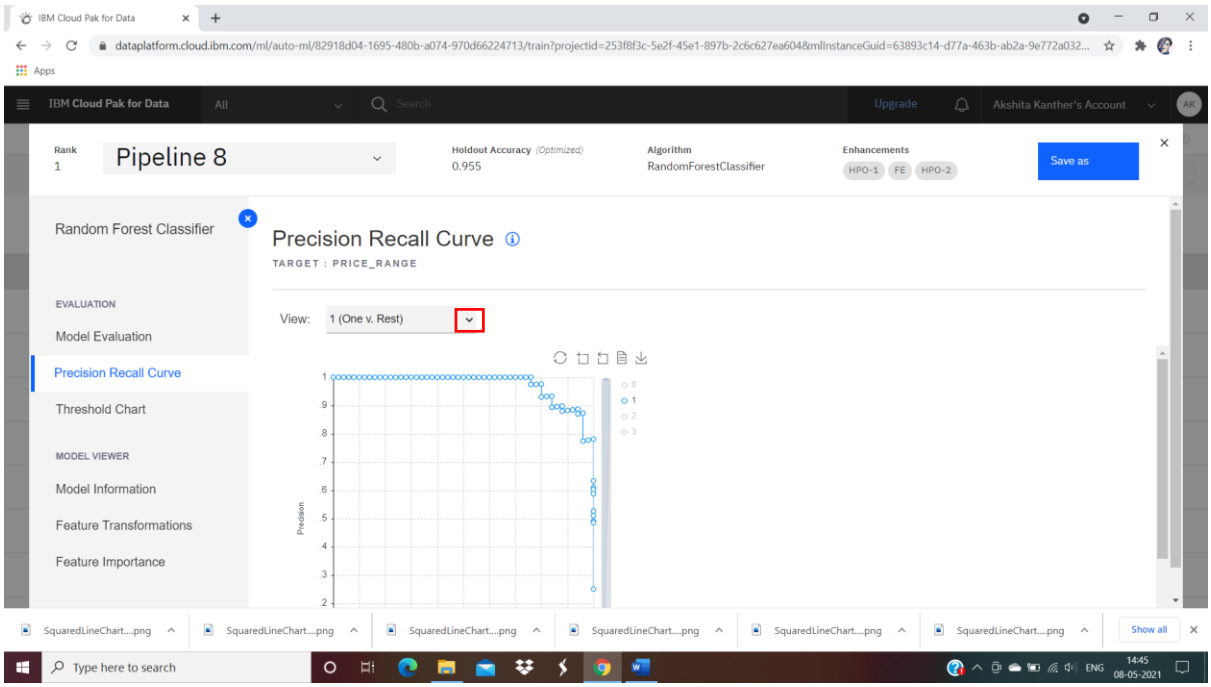
Step-26. Click on Precision Recall Curve on the left side and view it for Multi-Class view.



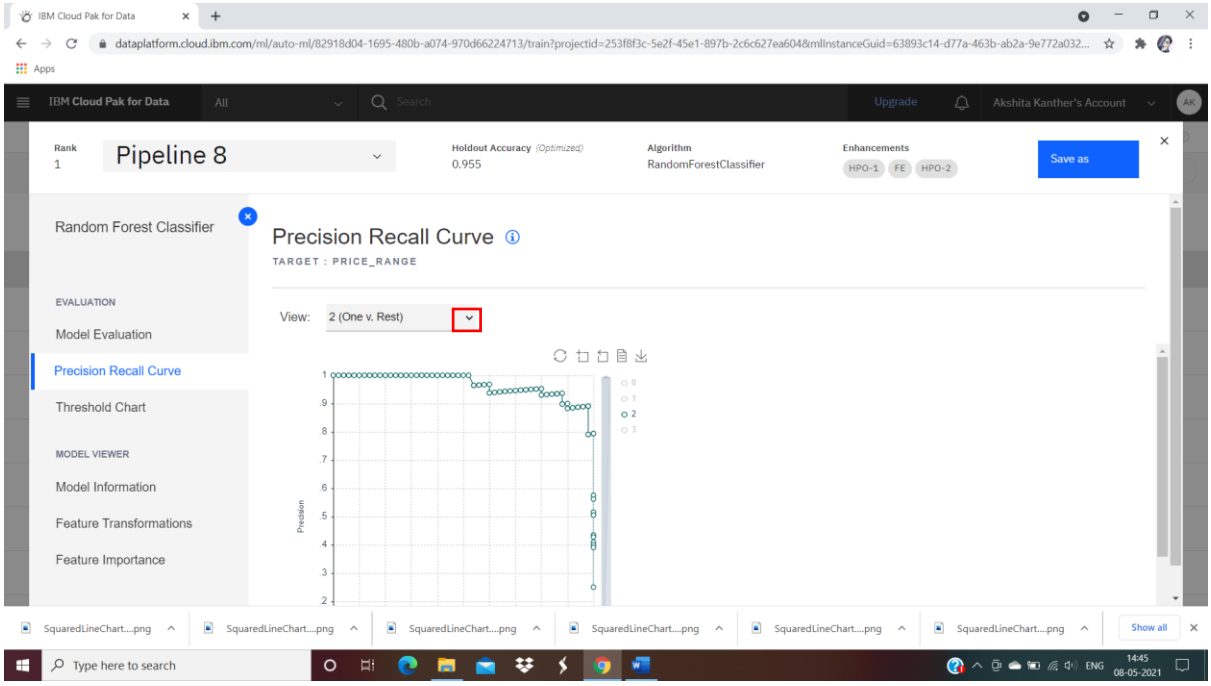
Step-27. Click on the dropdown which is present in front of the view and select 0(One v. Rest) .View the Precision-Recall curve for one class of the predicted column.



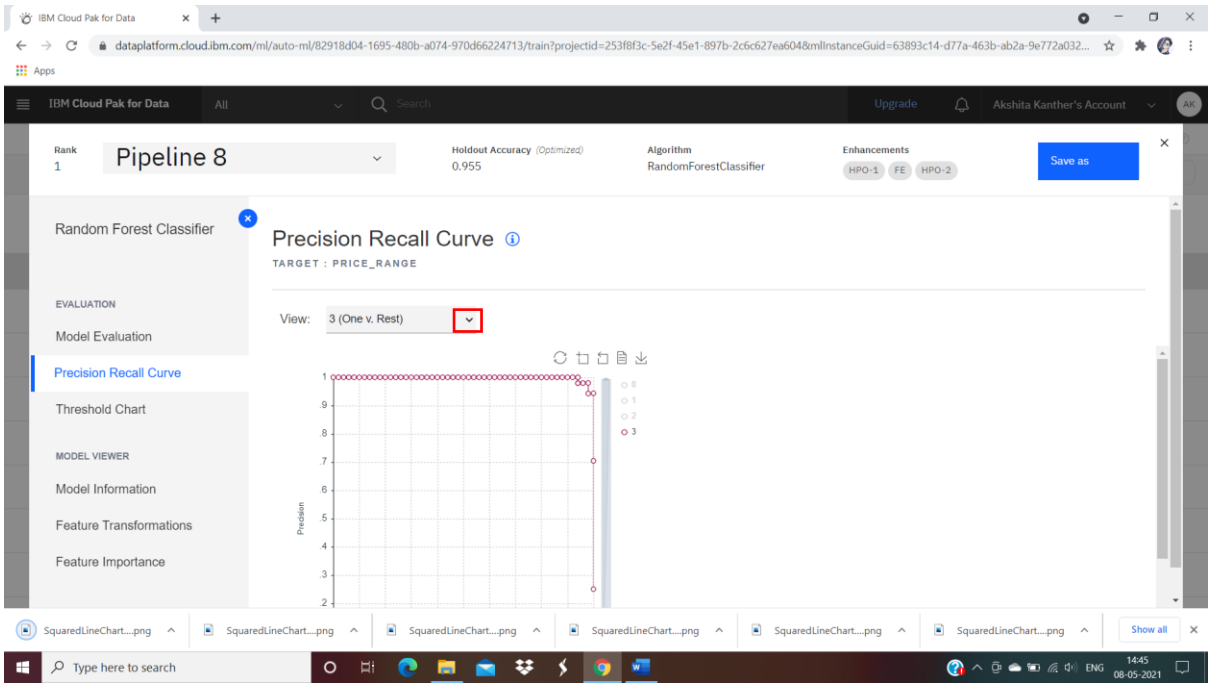
Step-28. Click on the dropdown which is present in front of the view and select 1(One v. Rest) .View the Precision-Recall curve for one class of the predicted column.

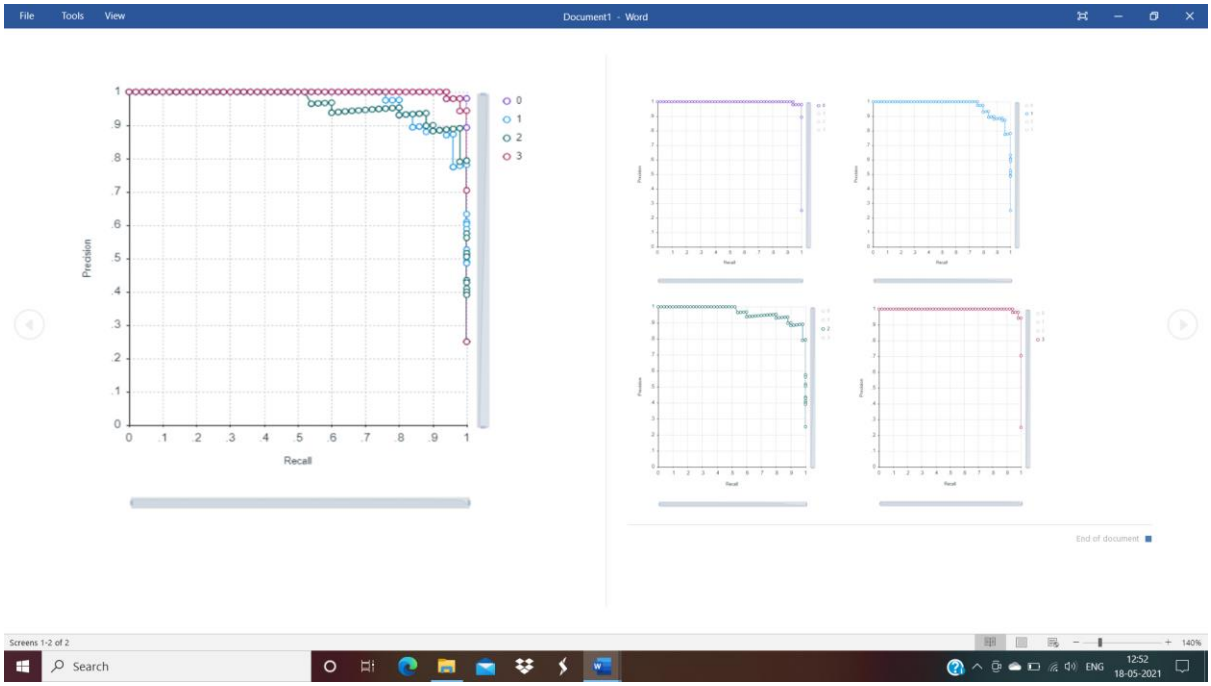


Step-29. Click on the dropdown which is present in front of the view and select 2(One v. Rest) .View the Precision-Recall curve for one class of the predicted column.

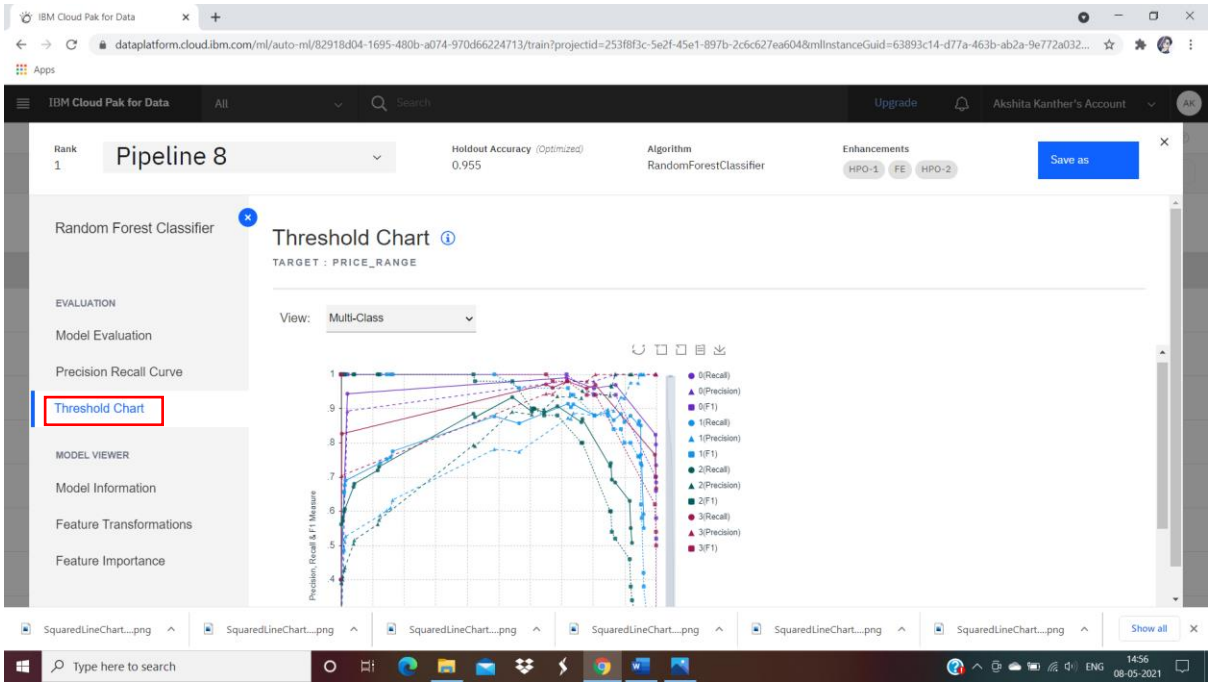


Step-30. Click on the dropdown which is present in front of the view and select 3(One v. Rest) .View the Precision-Recall curve for one class of the predicted column.

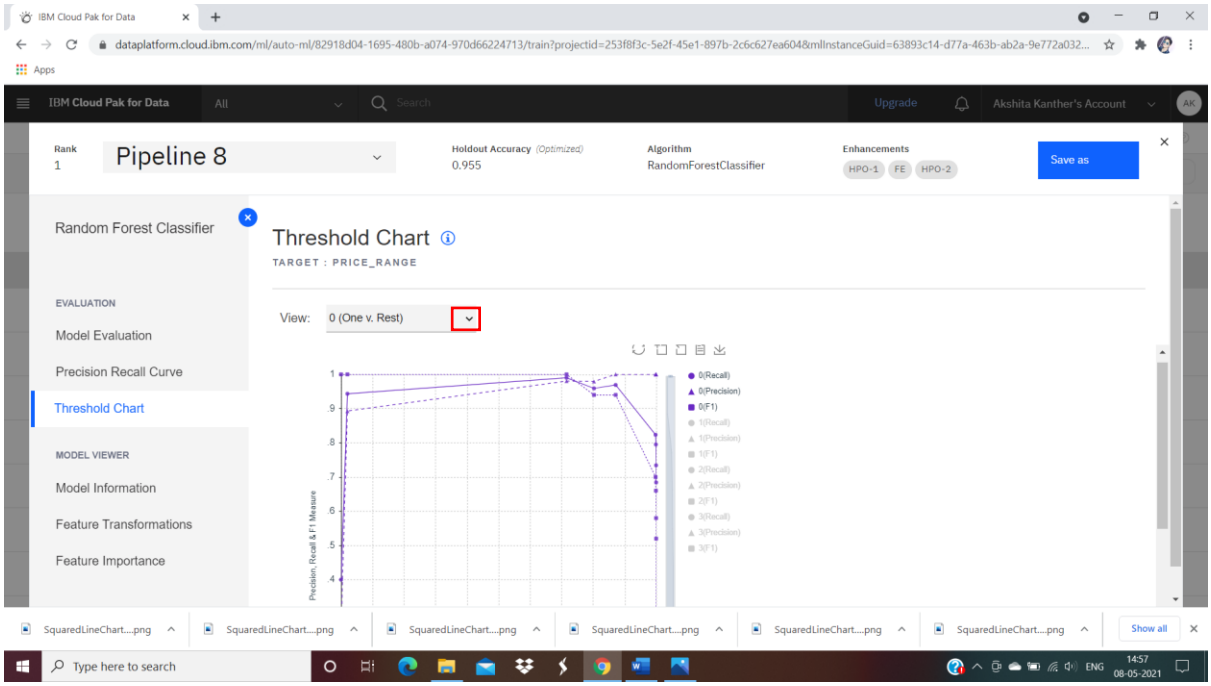




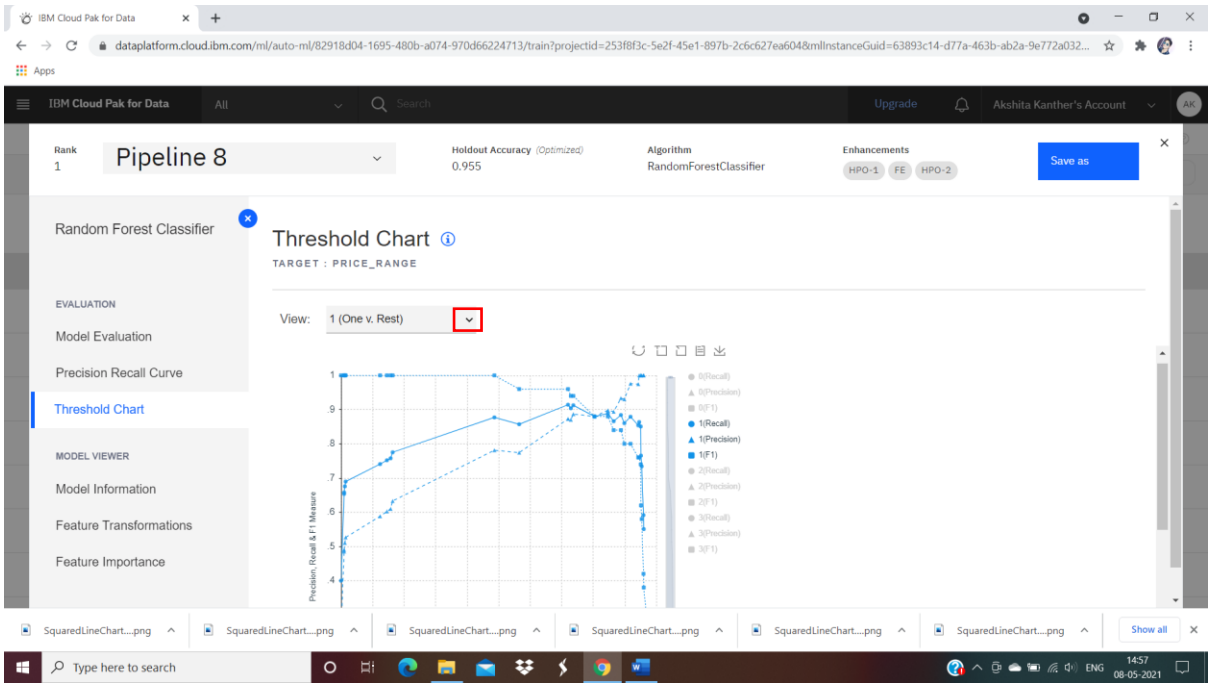
Step-31. Click on Threshold Chart on the left side and view it for Multi-Class view.



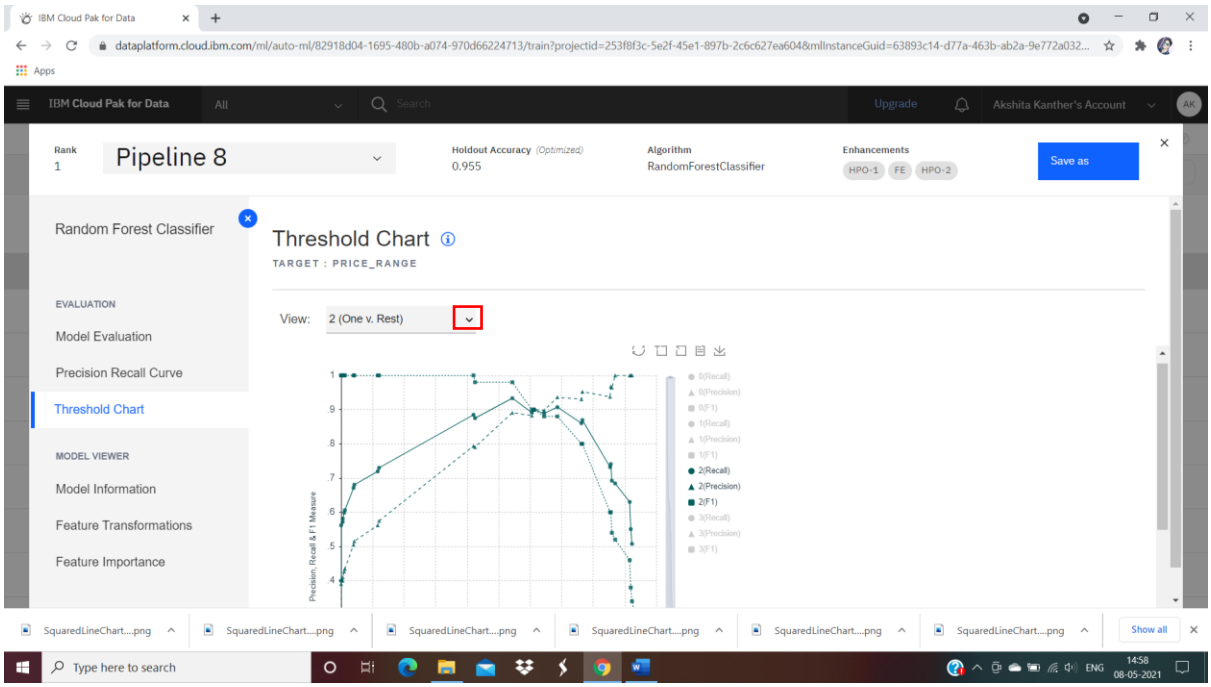
Step-32. Click on the dropdown which is present in front of the view and select 0(One v. Rest) .View the Threshold Chart for one class of the predicted column.



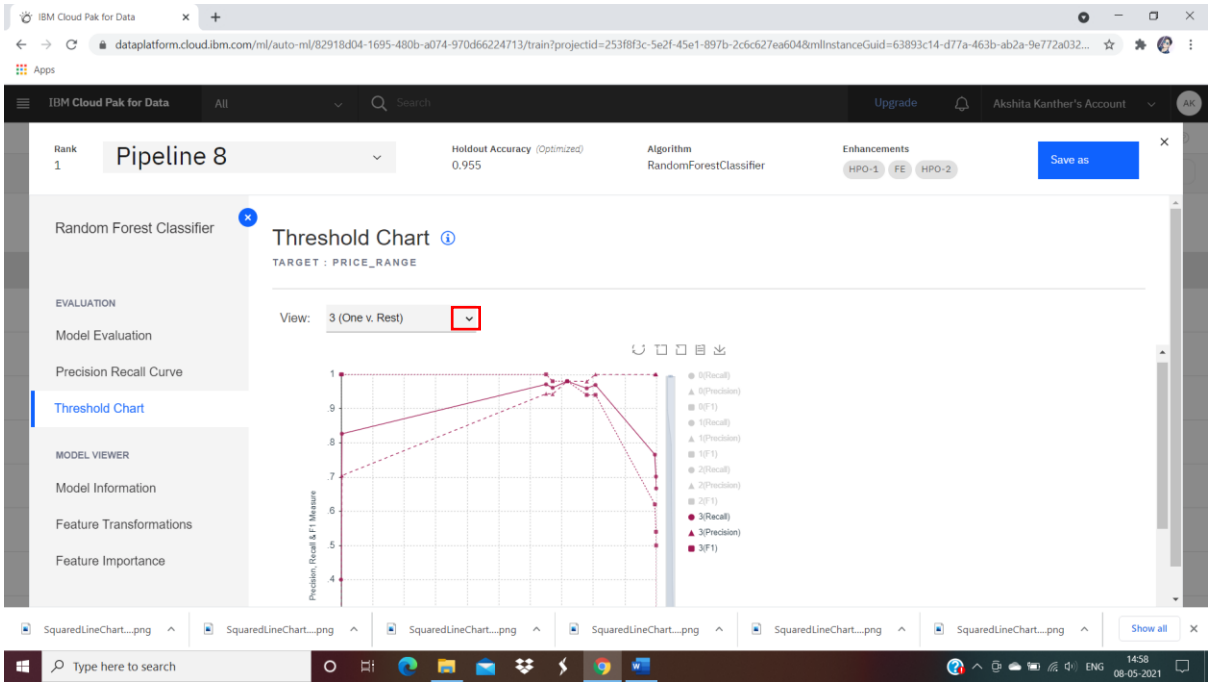
Step-33. Click on the dropdown which is present in front of the view and select 1(One v. Rest) .View the Threshold Chart for one class of the predicted column.

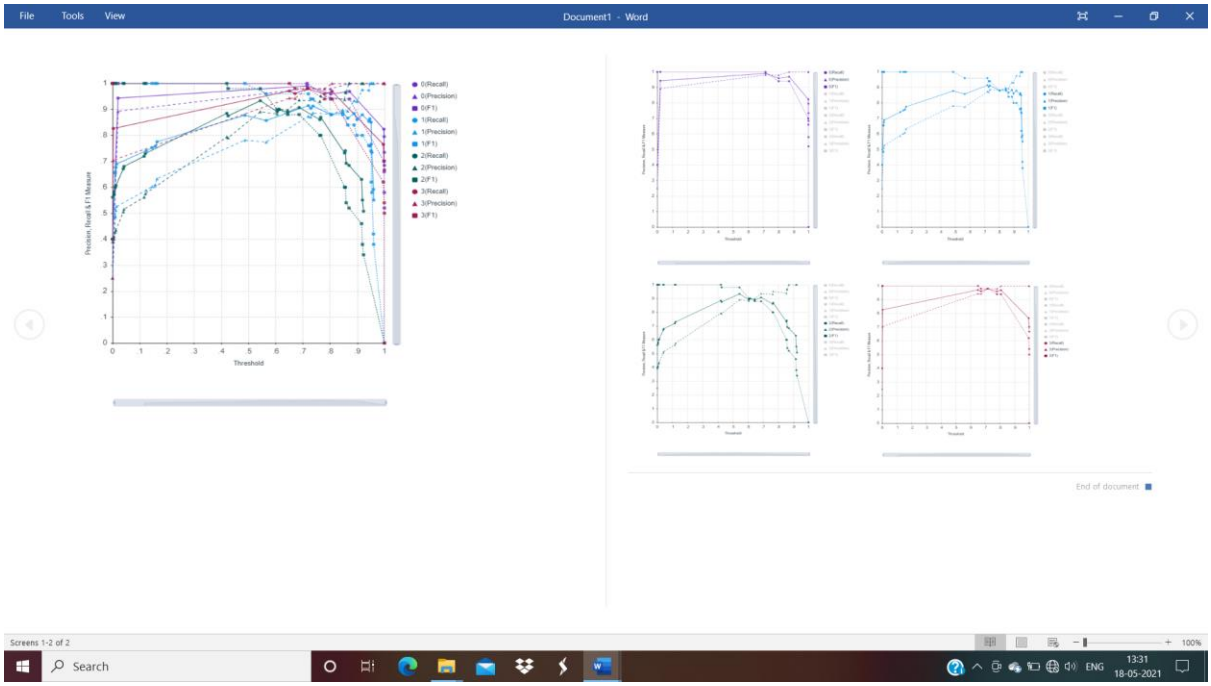


Step-34. Click on the dropdown which is present in front of the view and select 2(One v. Rest) .View the Threshold Chart for one class of the predicted column.

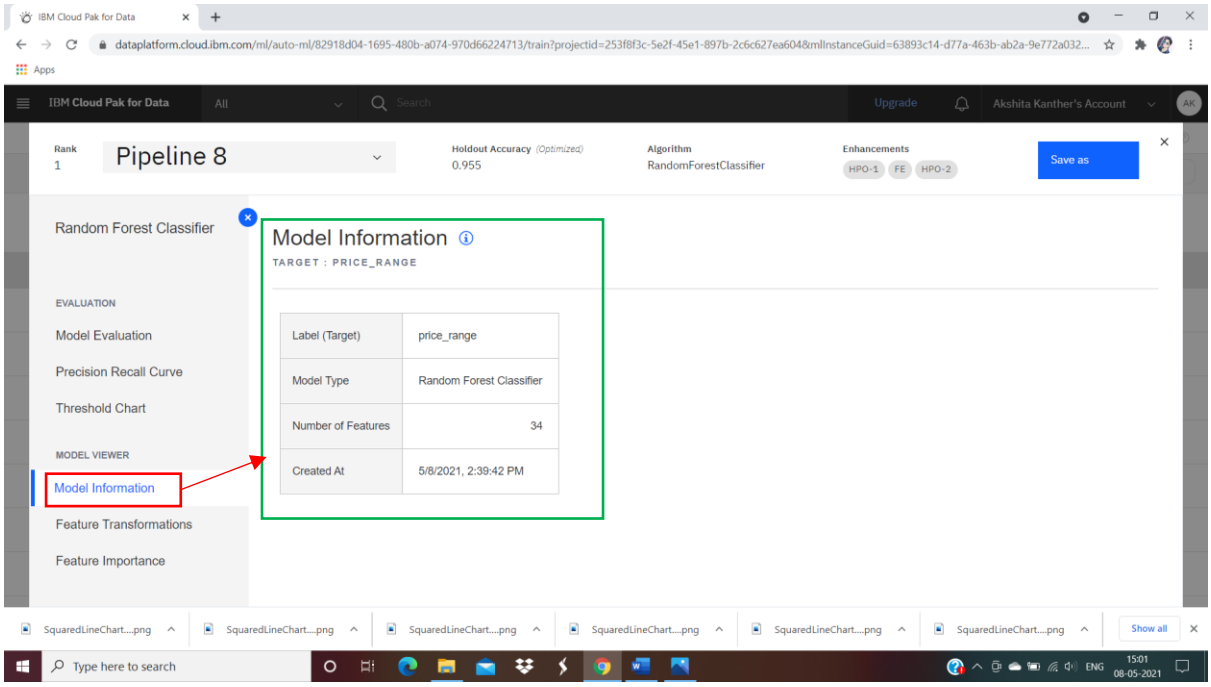


Step-35. Click on the dropdown which is present in front of the view and select 3(One v. Rest) .View the Threshold Chart for one class of the predicted column.

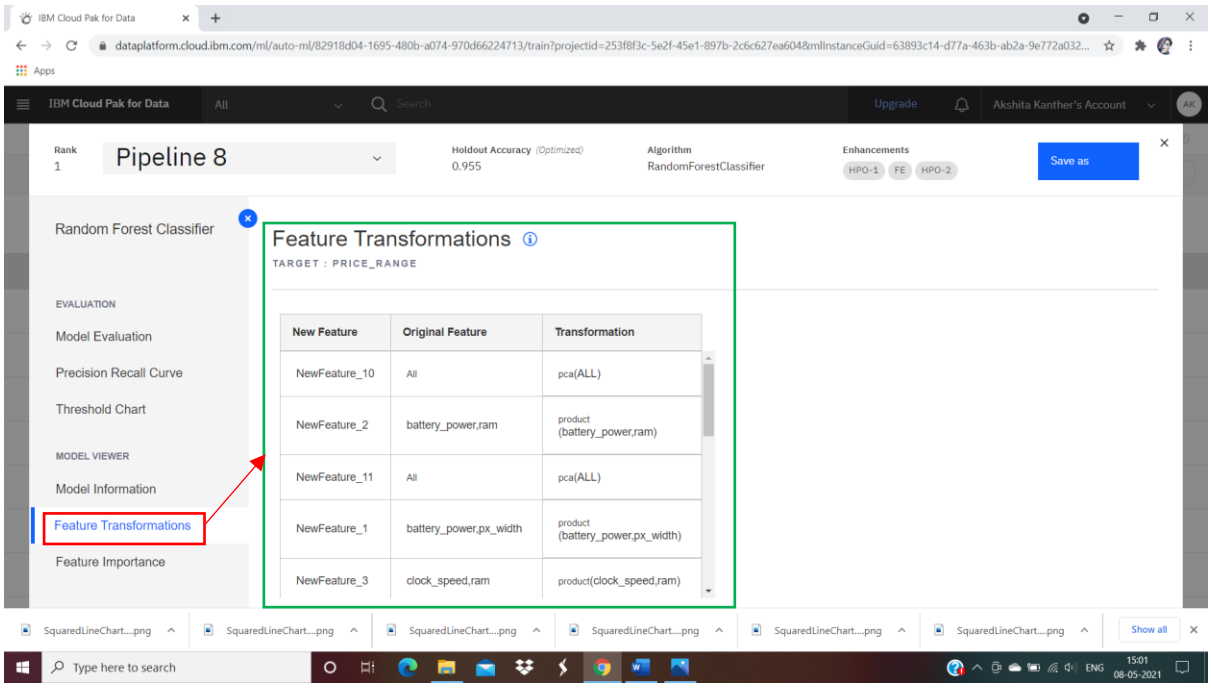




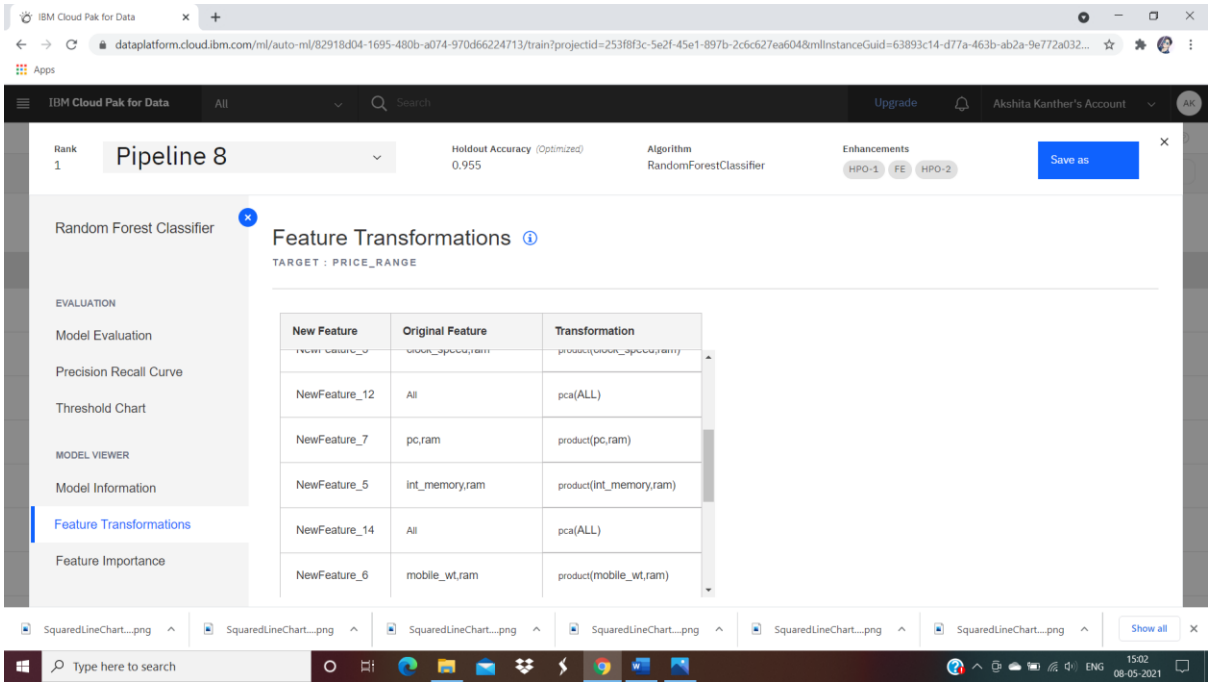
Step-36. Click on Model Information on the left side of the screen and view the Model Type, Number of Features, date of creation and Label.

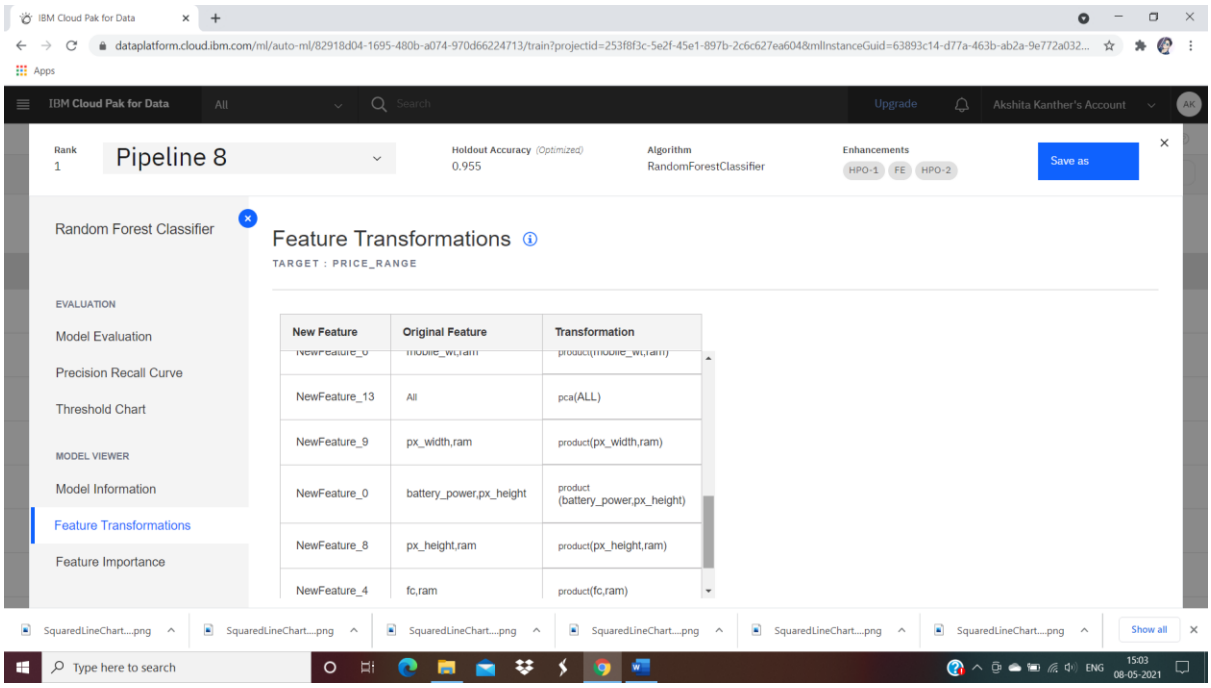


Step-37. Click on Feature Transformations on the left side of the screen and view the new features which are created or transformed by combining existing features or by applying Principal Component Analysis.

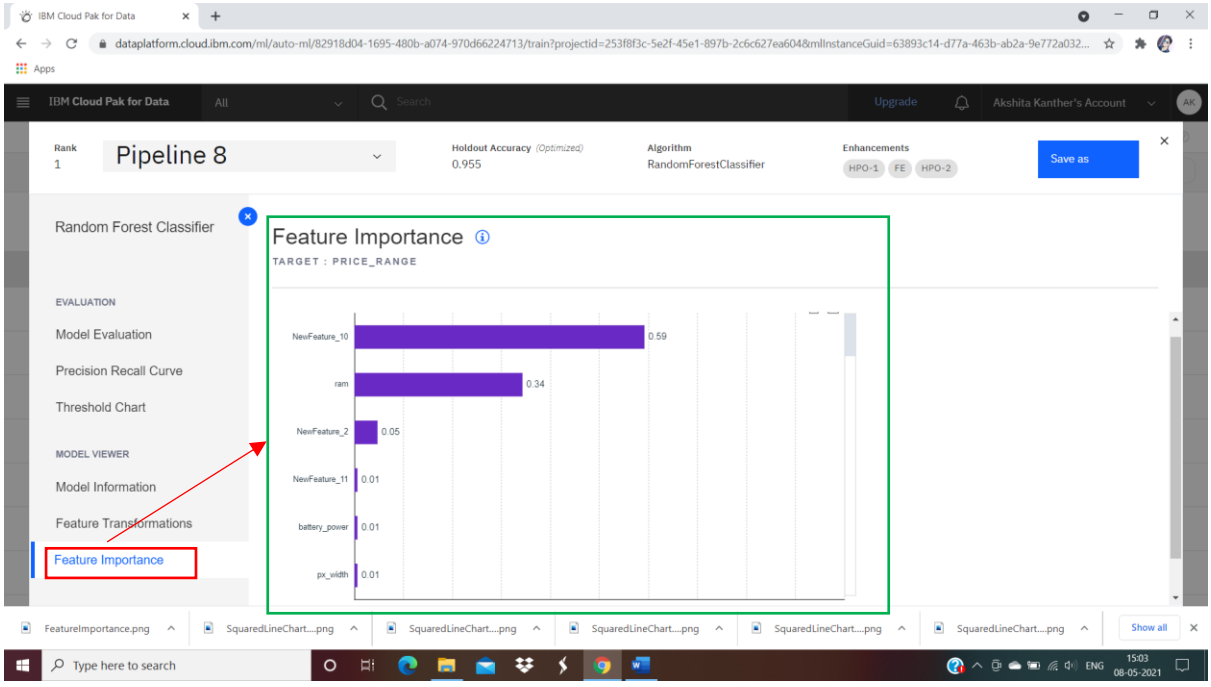


Step-38. Scroll down and observe all the feature transformations.

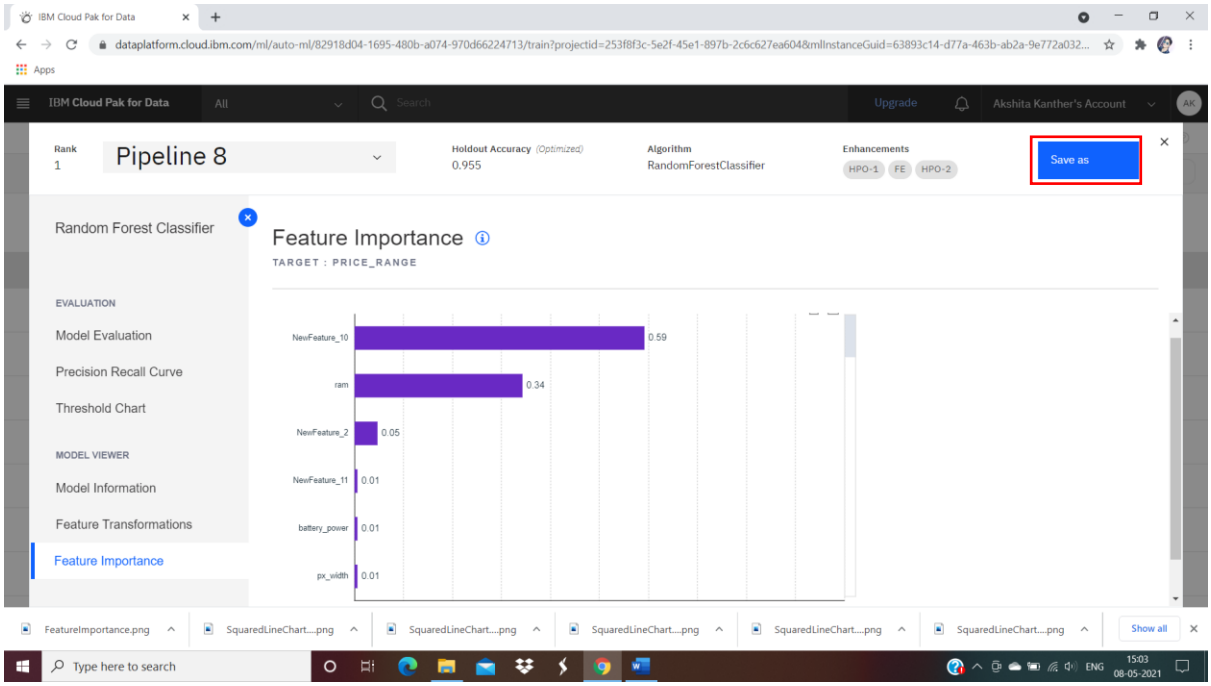


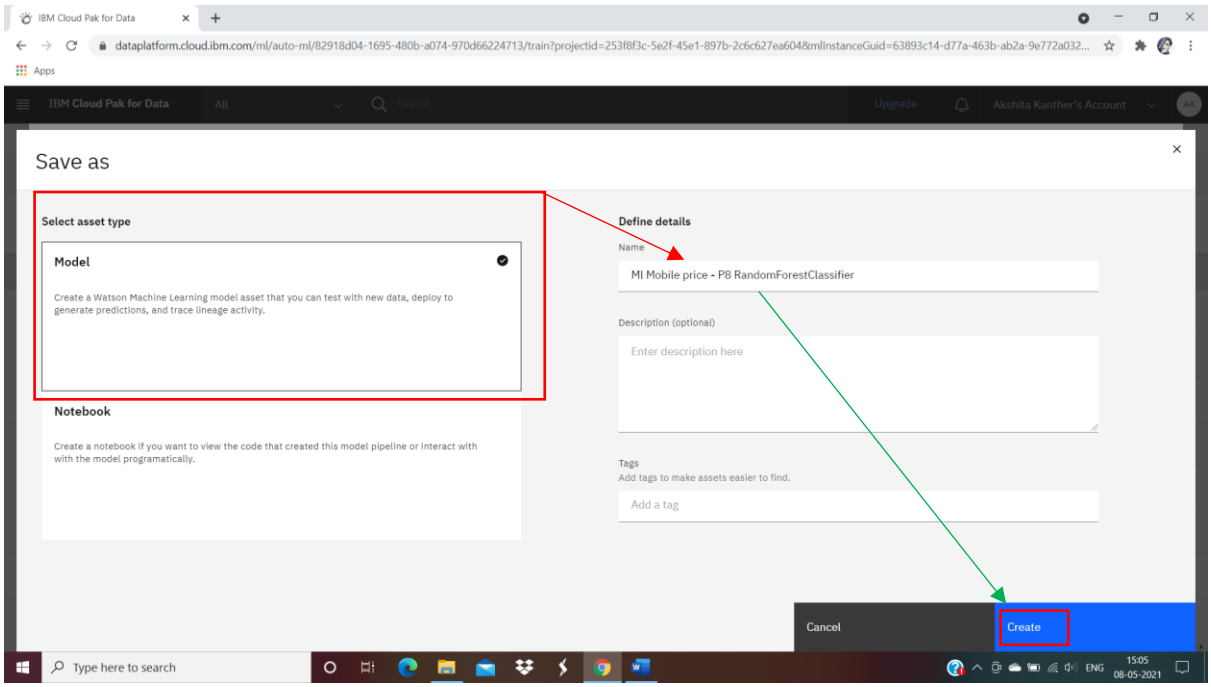


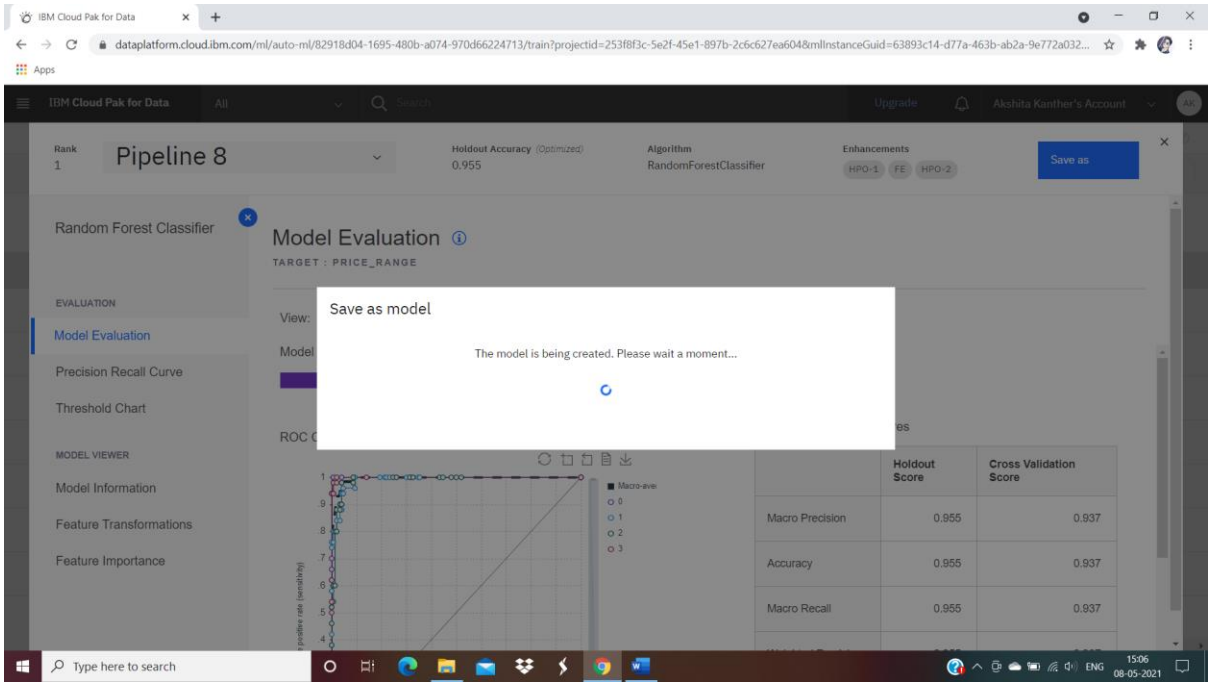
Step-39. Click on Feature Importance on the left side of the screen and observe the contribution of each feature in prediction.



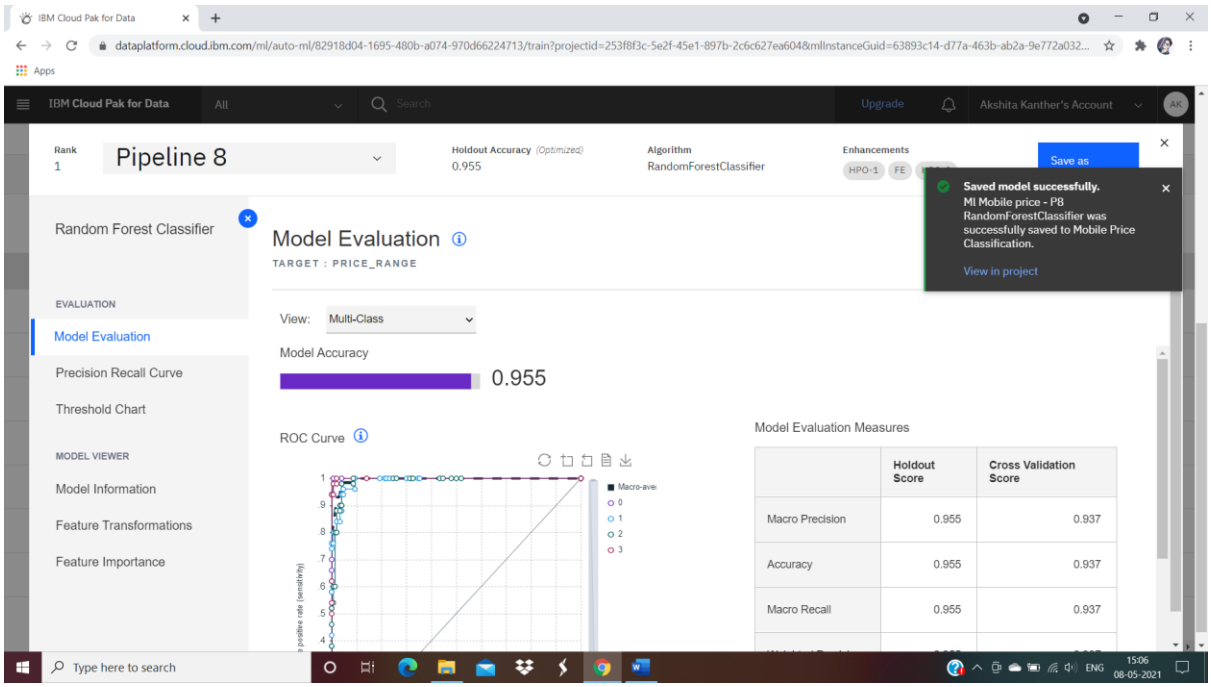
Step-40. Click on Save as on the top right corner. Select Model and then click on Create.


IBM Cloud Pak for Data
Rank
1
Pipeline 8
Holdout Accuracy (Optimized)
0.955
Algorithm
RandomForestClassifier
Enhancements
HPO-1
FE
HPO-2
Save as
Random Forest Classifier
EVALUATION
Model Evaluation
Precision Recall Curve
Threshold Chart
MODEL VIEWER
Model Information
Feature Transformations
Feature Importance
Feature Importance
TARGET : PRICE_RANGE
NewFeature_10
0.59
ram
0.34
NewFeature_2
0.05
NewFeature_11
0.01
battery_power
0.01
px_width
0.01

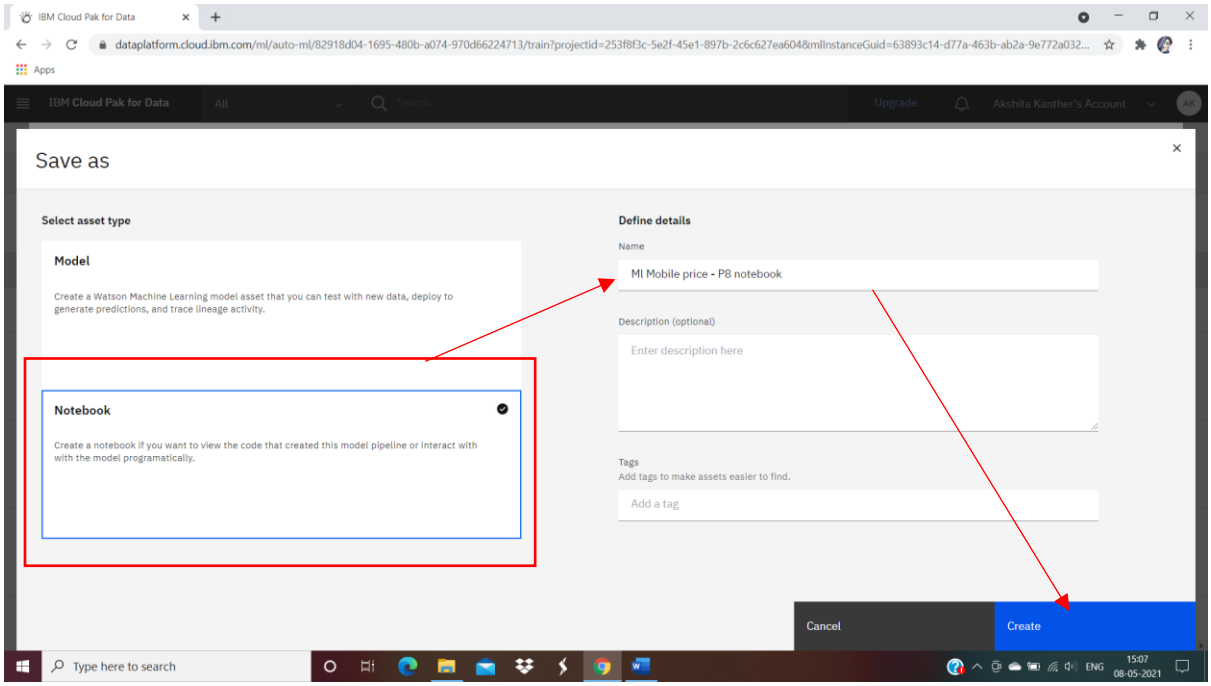

Save as
Select asset type
Model
Create a Watson Machine Learning model asset that you can test with new data, deploy to generate predictions, and trace lineage activity.
Notebook
Create a notebook if you want to view the code that created this model pipeline or interact with with the model programatically.
Define details
Name
MI Mobile price - P8 RandomForestClassifier
Description (optional)
Enter description here
Tags
Add tags to make assets easier to find.
Add a tag
Cancel
Create

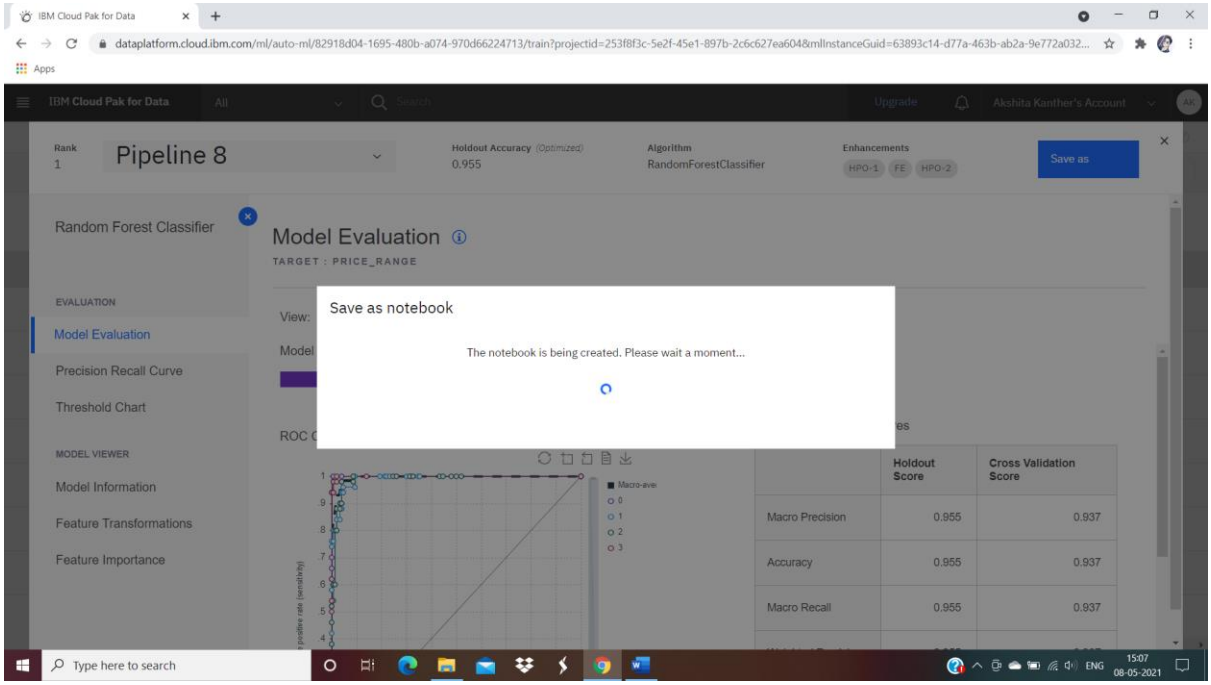


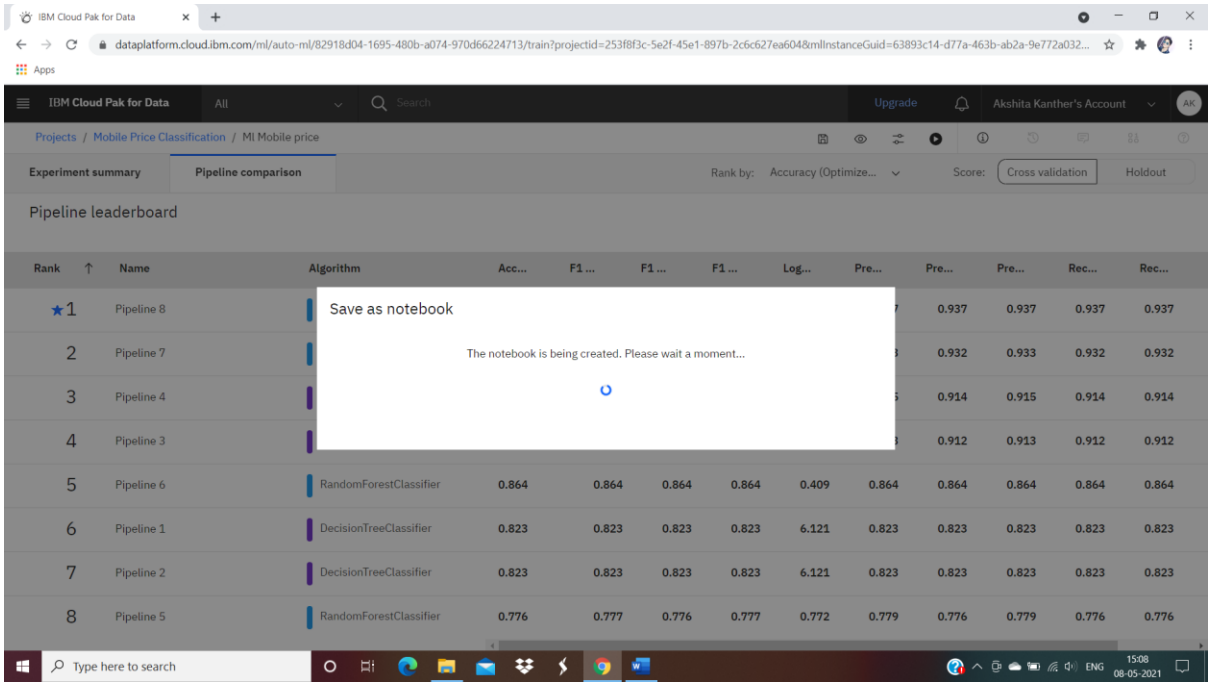
Step-41. A dialog box will pop up, it will display a message –"Saved the model successfully". Close the pop up and again click on Save as.



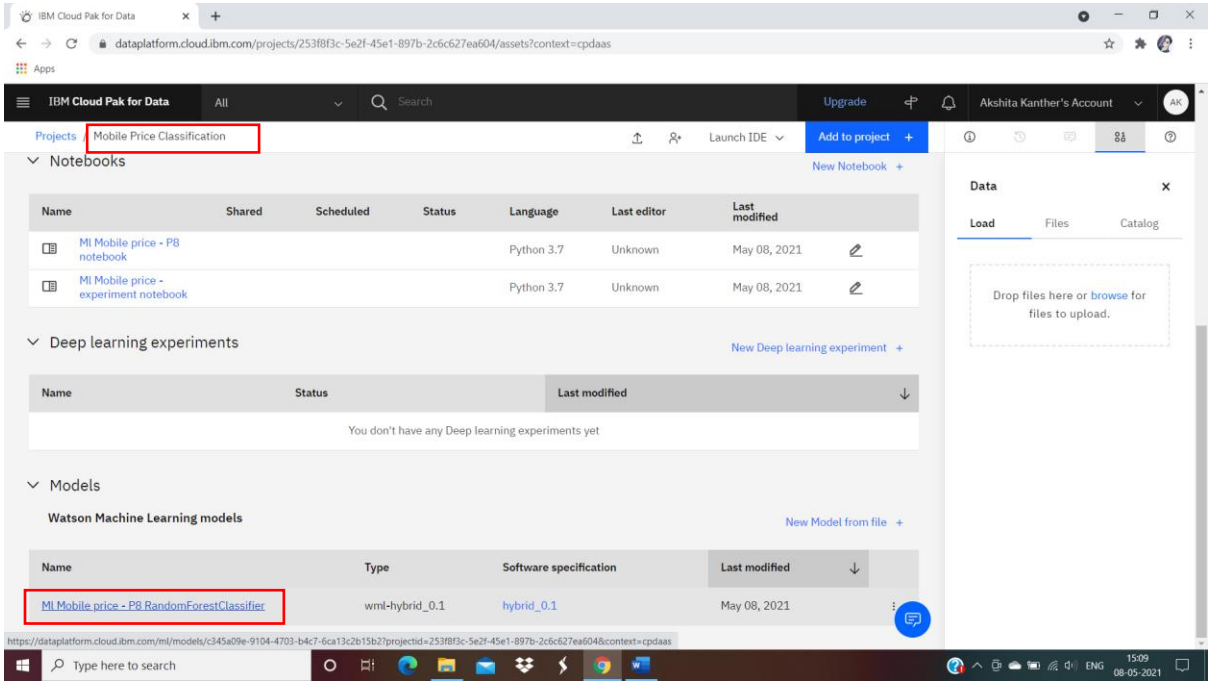
Step-42. Select Notebook and click on Create.


IBM Cloud Pak for Data
Save as
Select asset type
Model
Create a Watson Machine Learning model asset that you can test with new data, deploy to generate predictions, and trace lineage activity.
Notebook
Create a notebook if you want to view the code that created this model pipeline or interact with with the model programatically.
Define details
Name
MI Mobile price - P8 notebook
Description (optional)
Enter description here
Tags
Add tags to make assets easier to find.
Add a tag
Cancel
Create

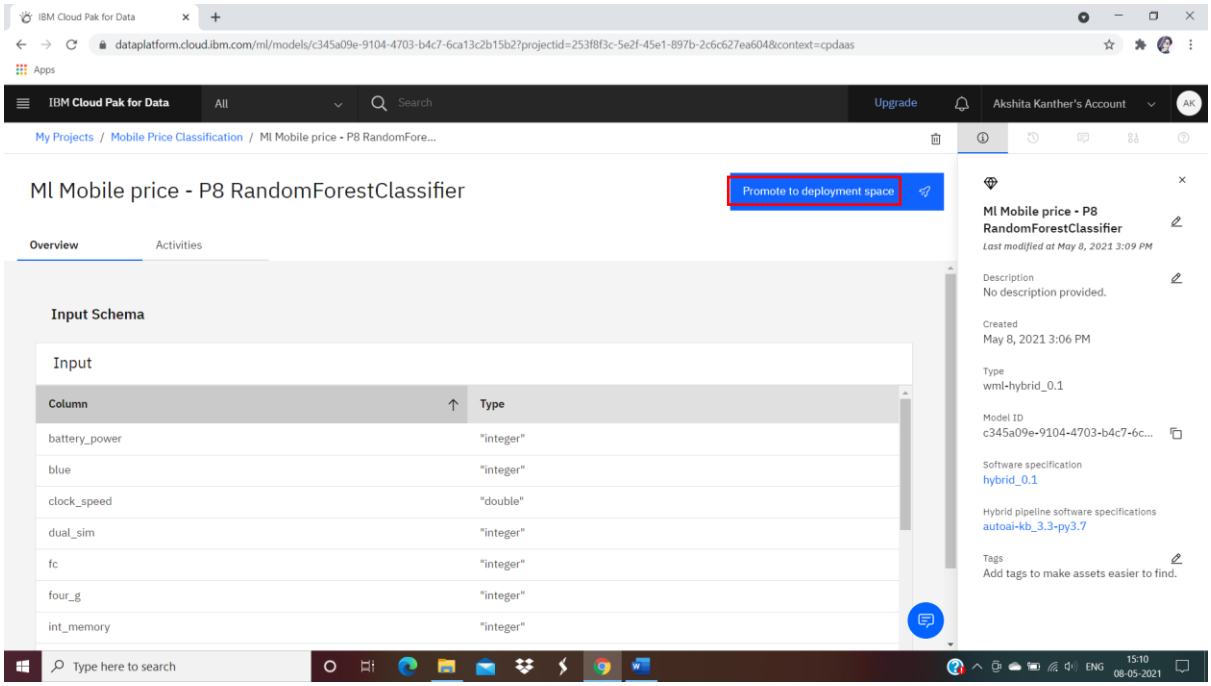

IBM Cloud Pak for Data
Rank 1
Pipeline 8
Holdout Accuracy (Optimized) 0.955
Algorithm RandomForestClassifier
Enhancements HPO-1 FE HPO-2
Save as
Random Forest Classifier
Model Evaluation
TARGET : PRICE_RANGE
EVALUATION
Model Evaluation
Precision Recall Curve
Threshold Chart
MODEL VIEWER
Model Information
Feature Transformations
Feature Importance
Save as notebook
The notebook is being created. Please wait a moment...
Holdout Score
Cross Validation Score
Macro Precision 0.955 0.937
Accuracy 0.955 0.937
Macro Recall 0.955 0.937



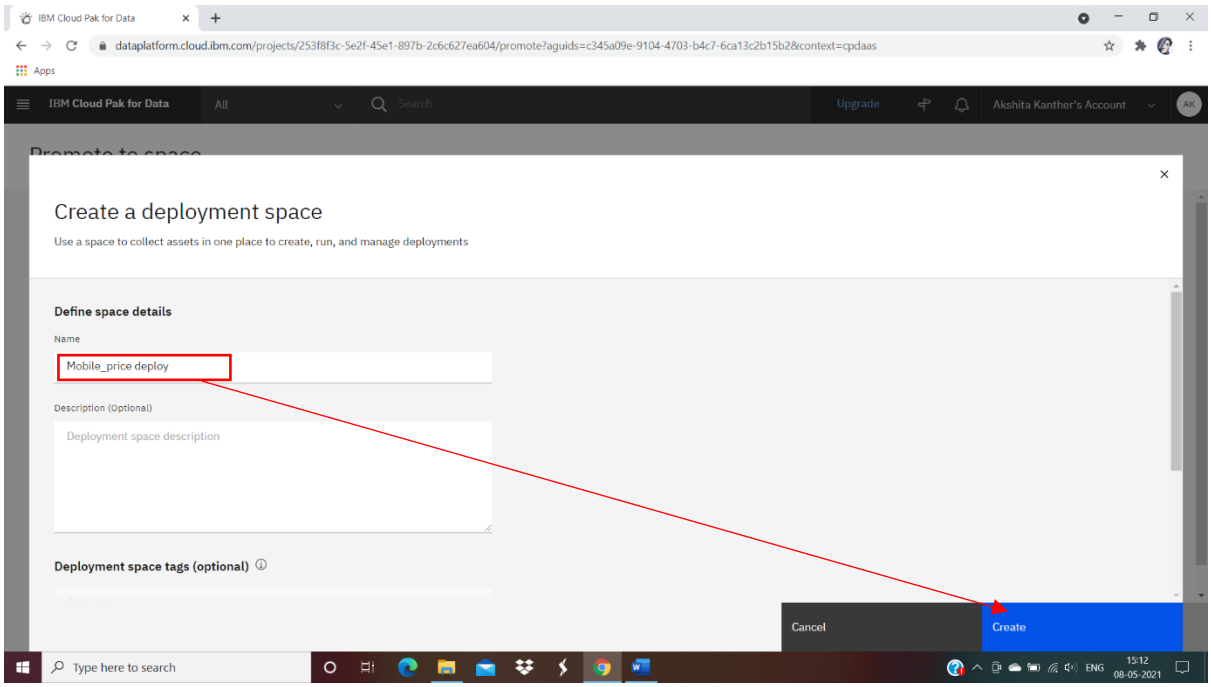
Step-43. Click on project name on the top (breadcrumb link on the top). And scroll down. Then click on the model name under Models.

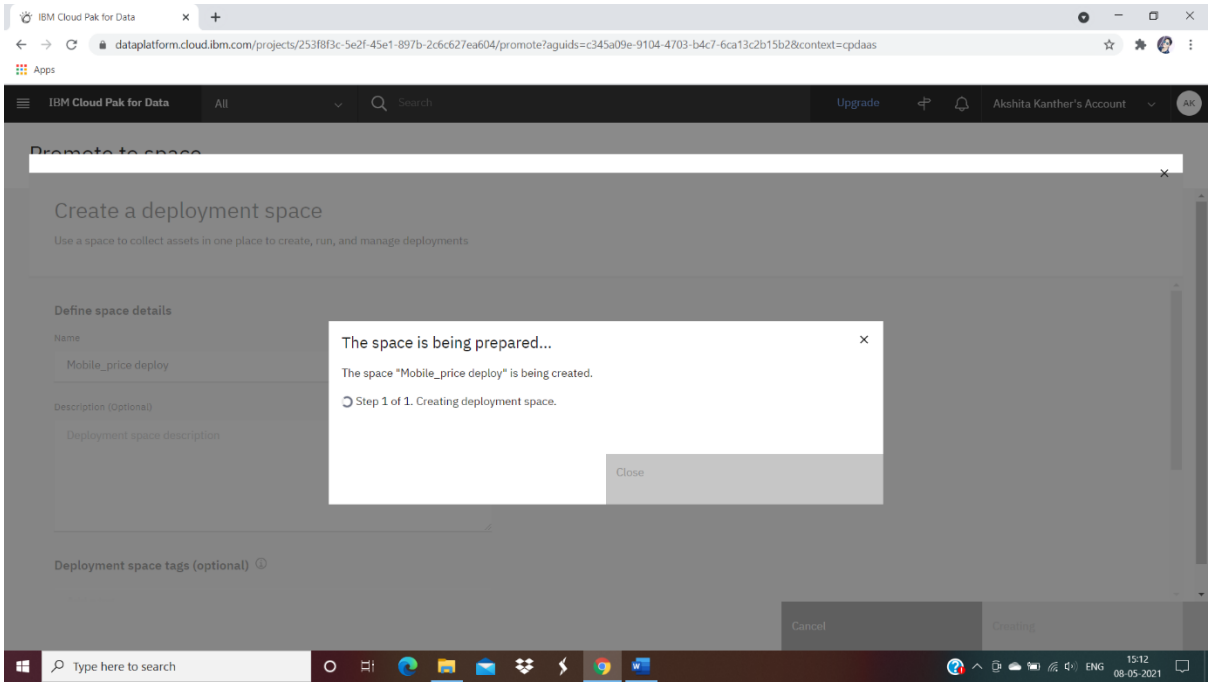


Step-44. Click on Promote to deployment space.

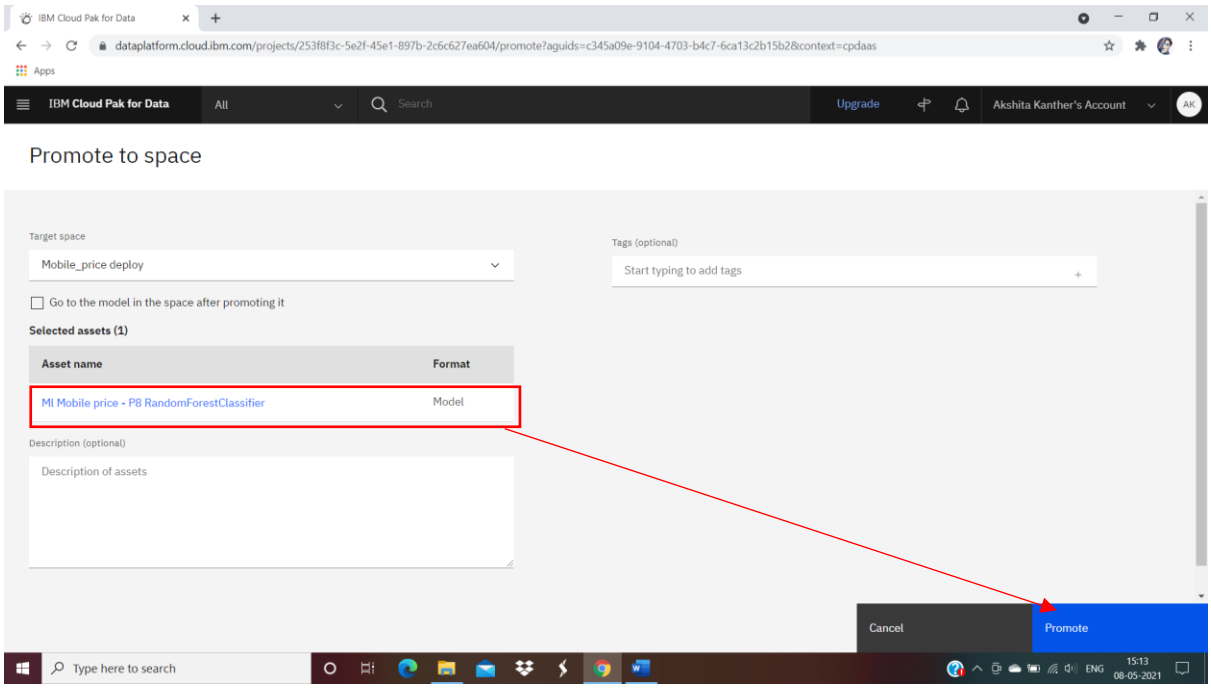


Step-45. Enter the name for deployment under Define space details and then click on Create.

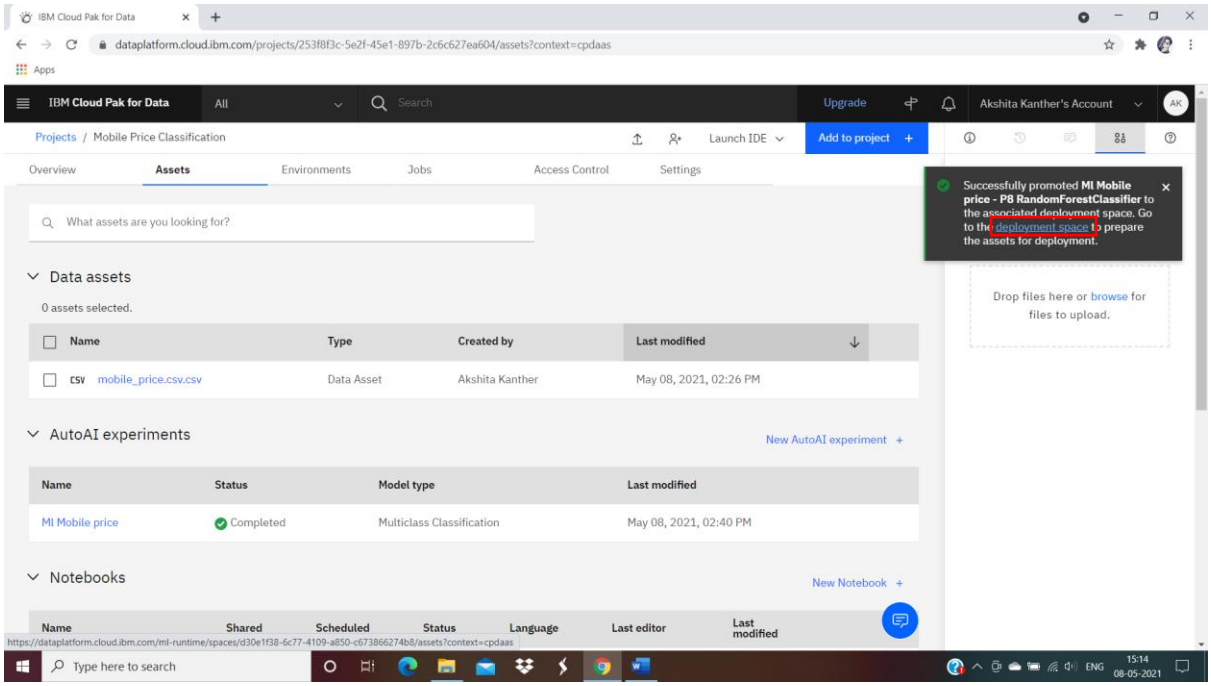




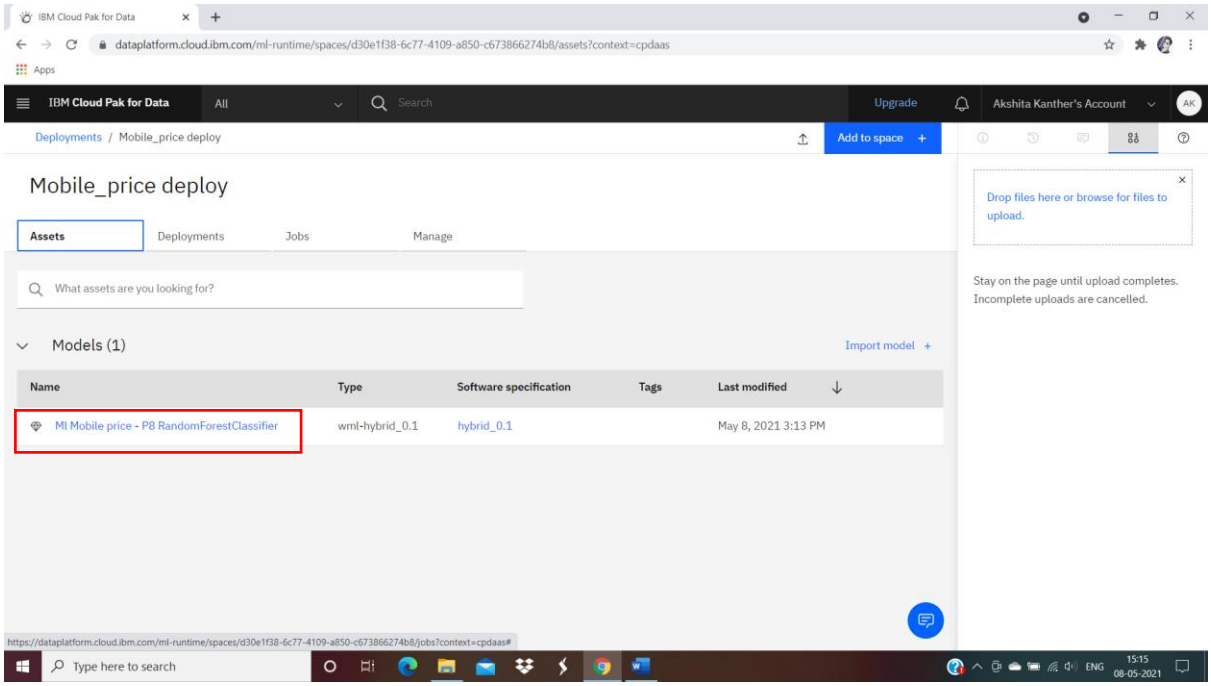
Step-46. Select a Target space from the dropdown box and then click on Promote.



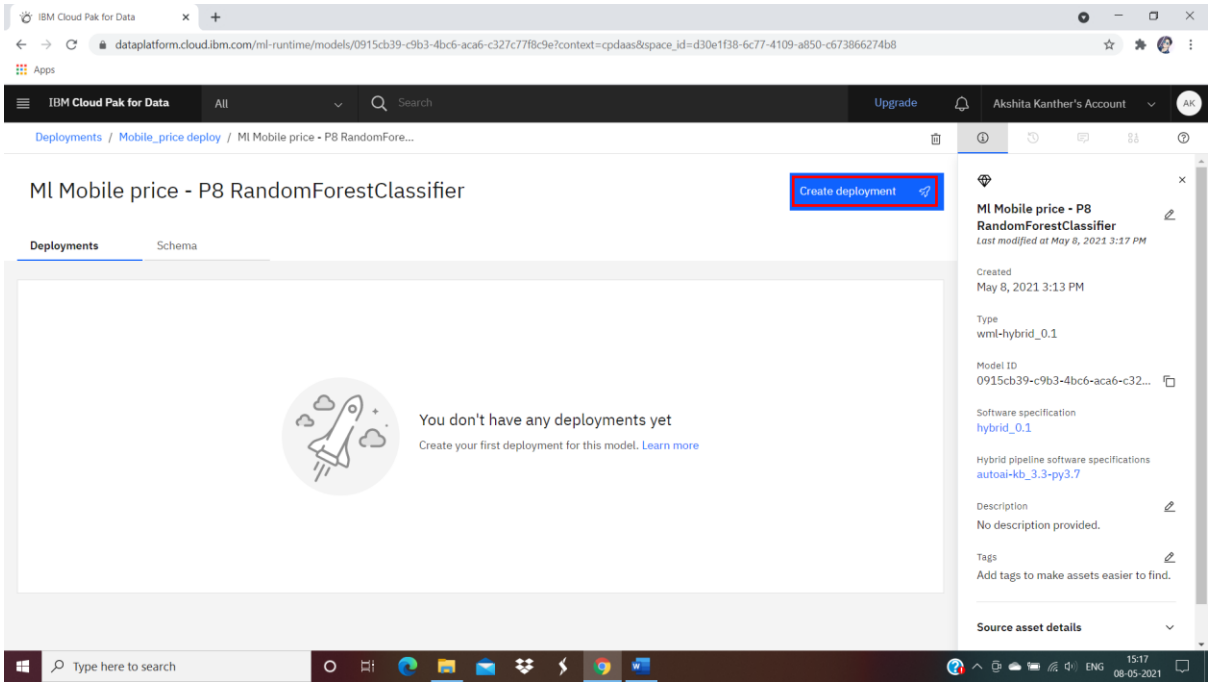
Step-47. A dialog box containing message of “Successfully promoted” will be displayed. Click on deployment space link in that message of the dialog box.



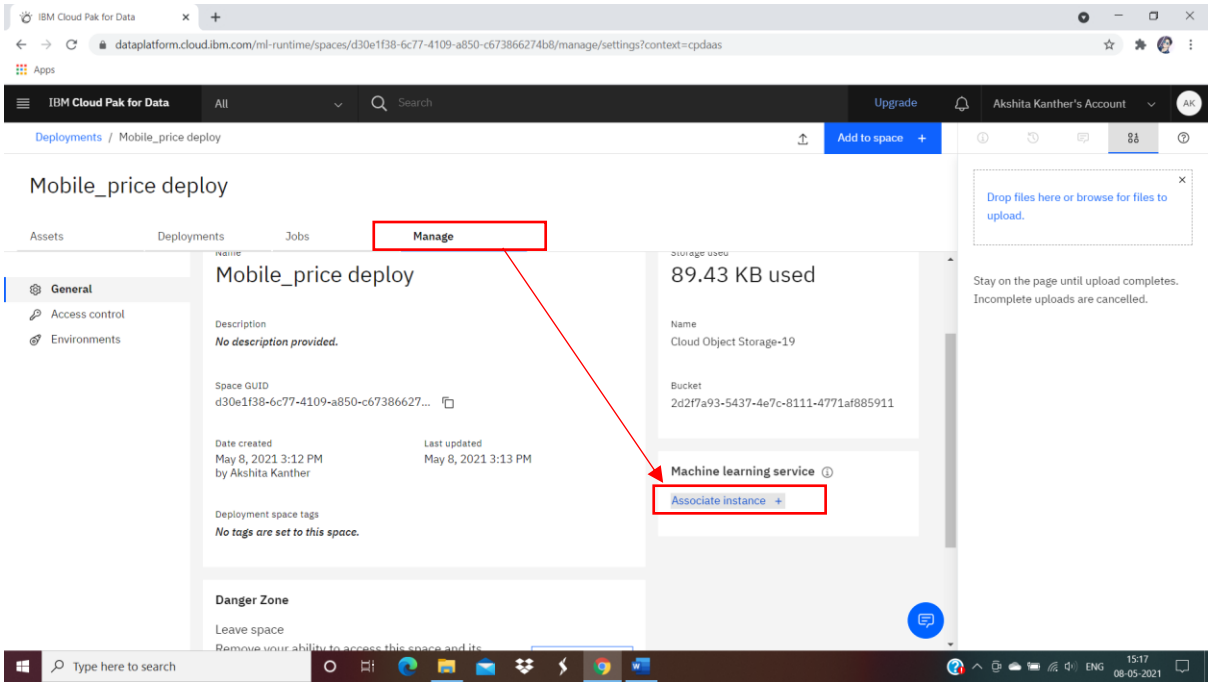
Step-48. Click on model name under Models (1) in the Assets tab.



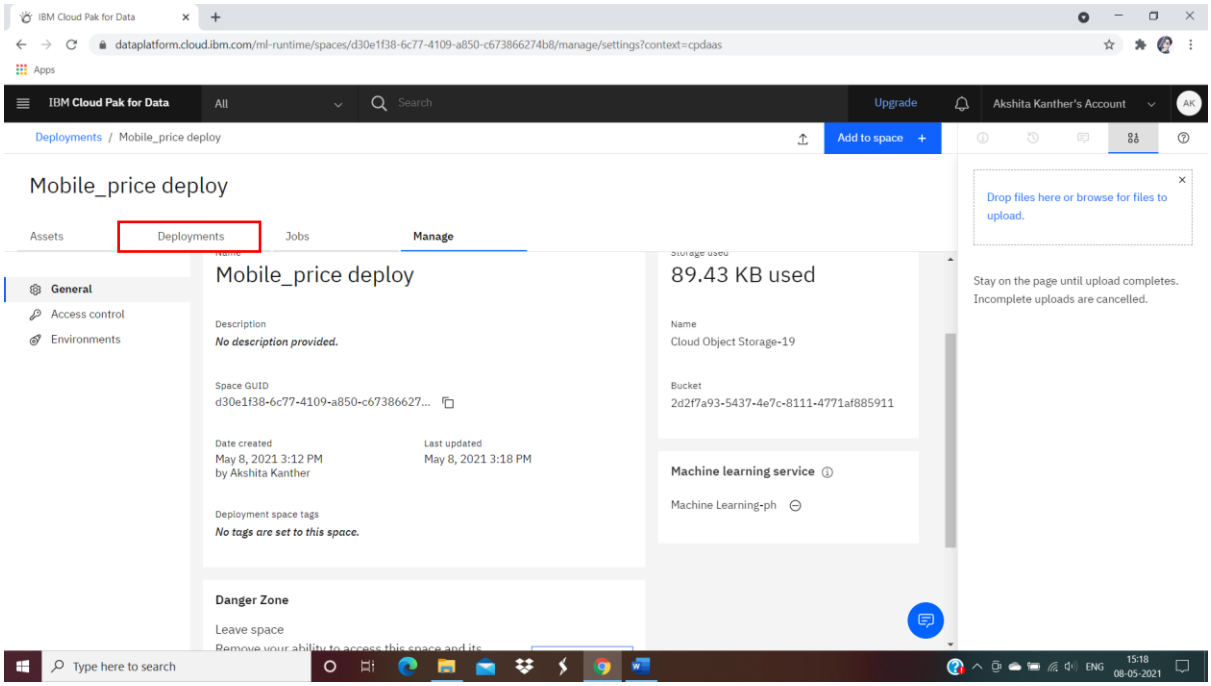
Step-49. Click on Create deployment.



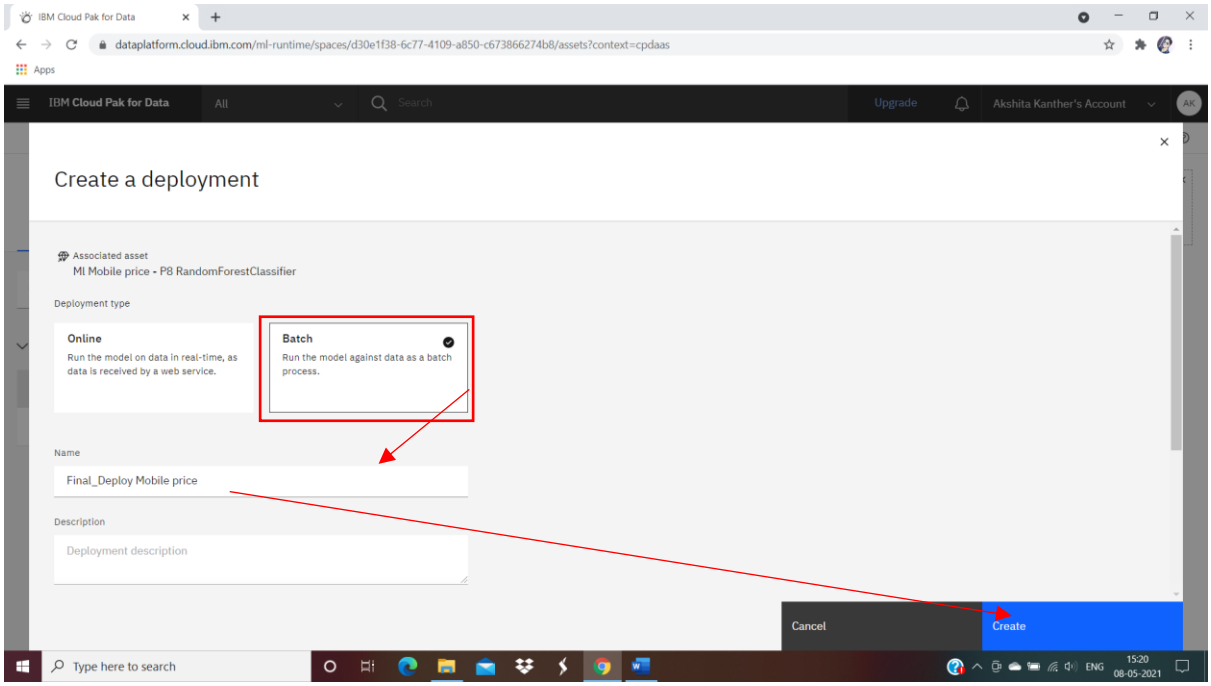
Step-50. Click on Manage. And then click on Associate instance + .



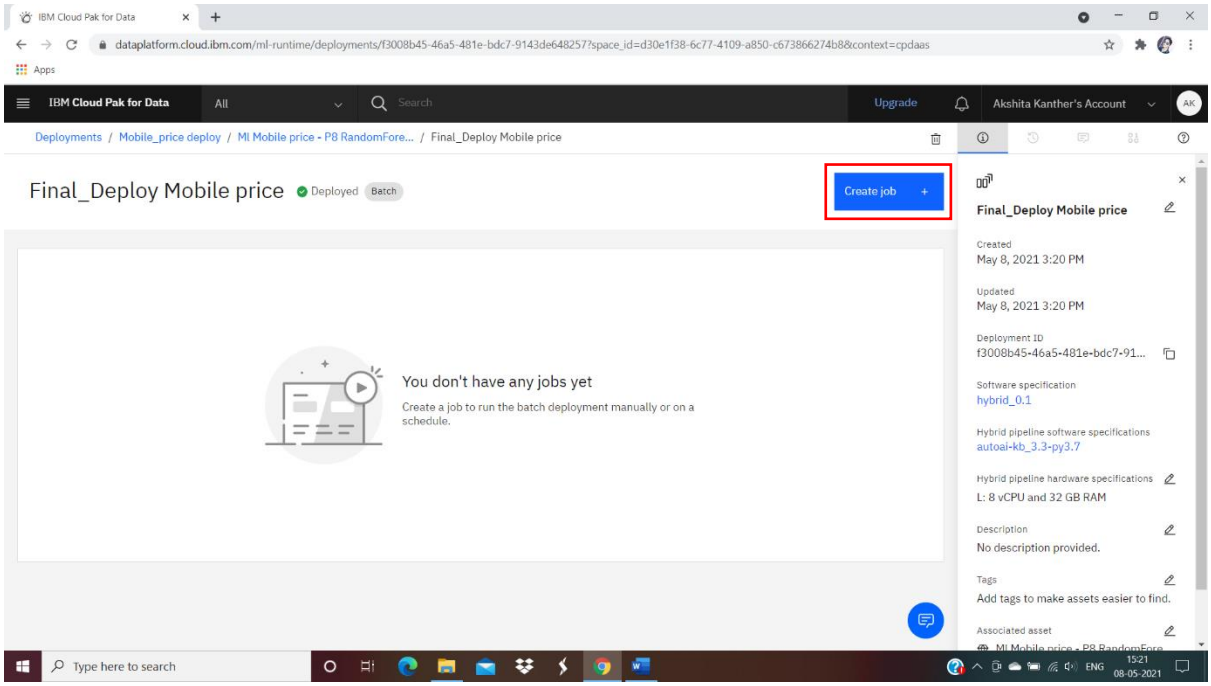
Step-51. After Machine learning service has been associated, switch to Deployments tab.



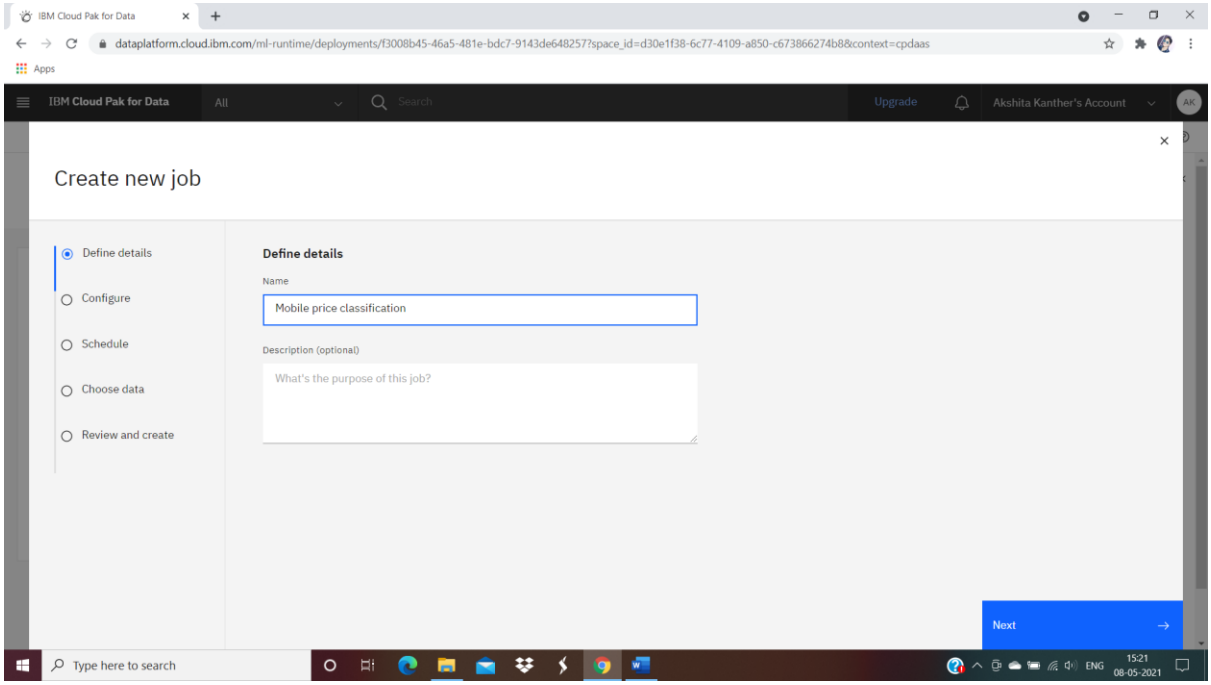
Step-52. Select Batch as Deployment type and then give a name and then click on Create.



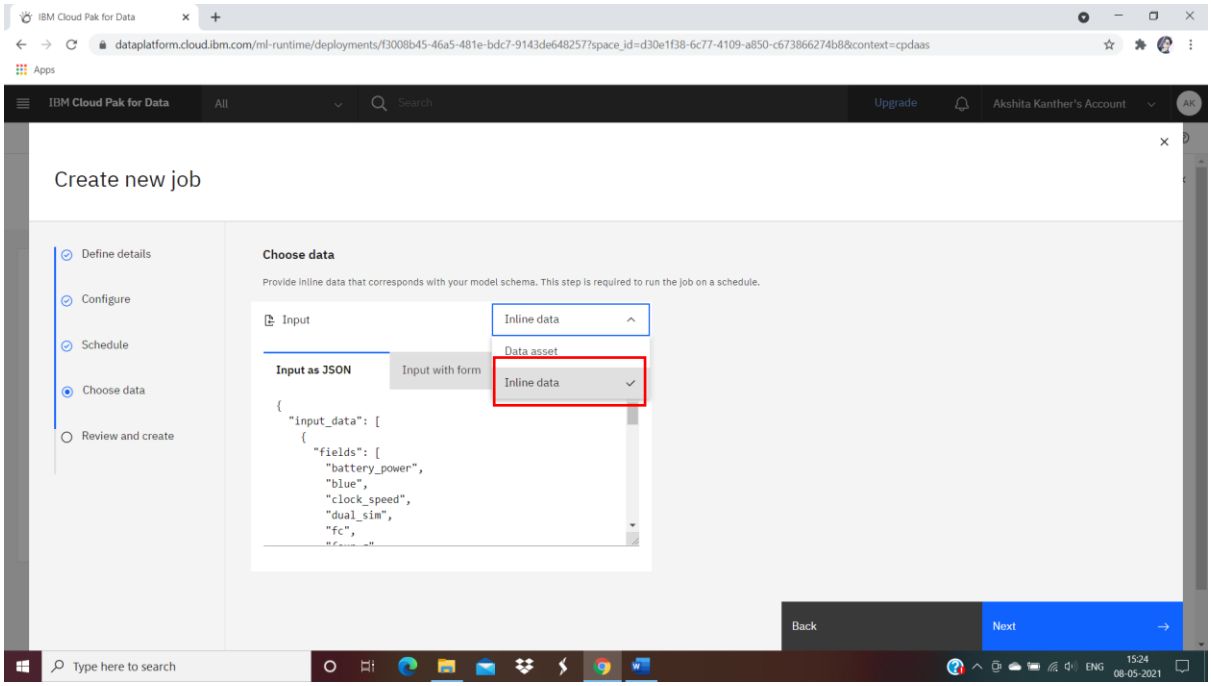
Step-53. Click on Create job + to create a job as we don’t have any jobs as of now.



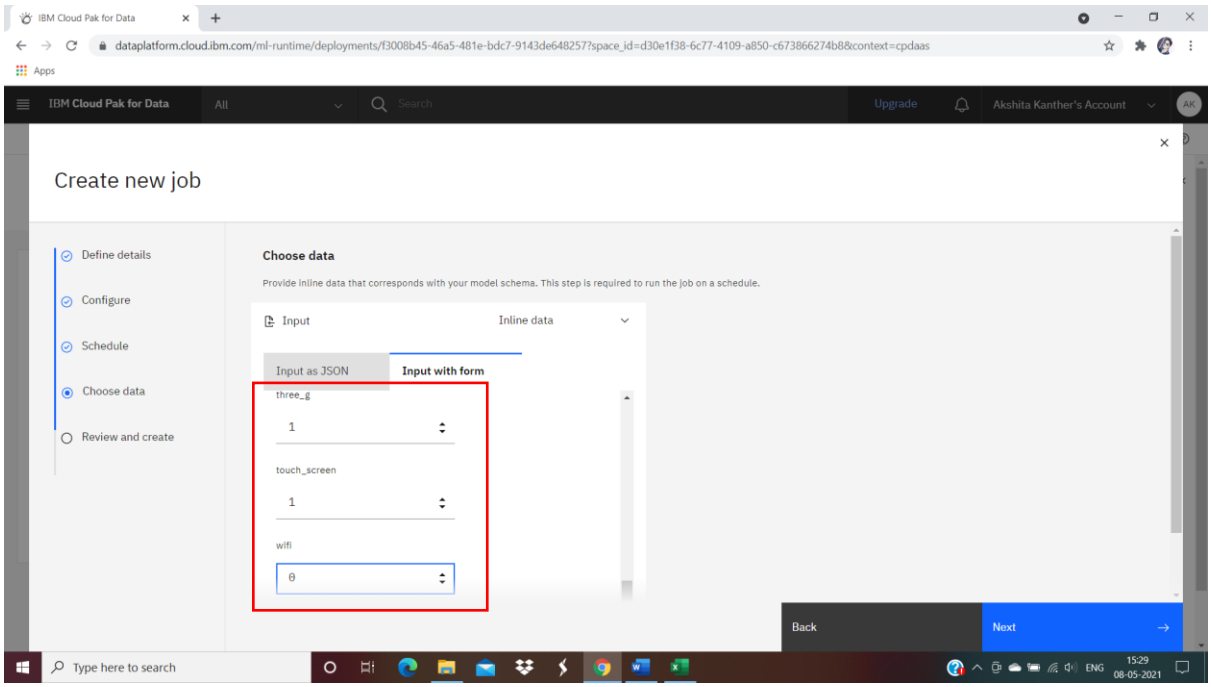
Step-54. Enter job name under Name column of Define details and then click on Next repeatedly for Define details, Configure and Schedule.



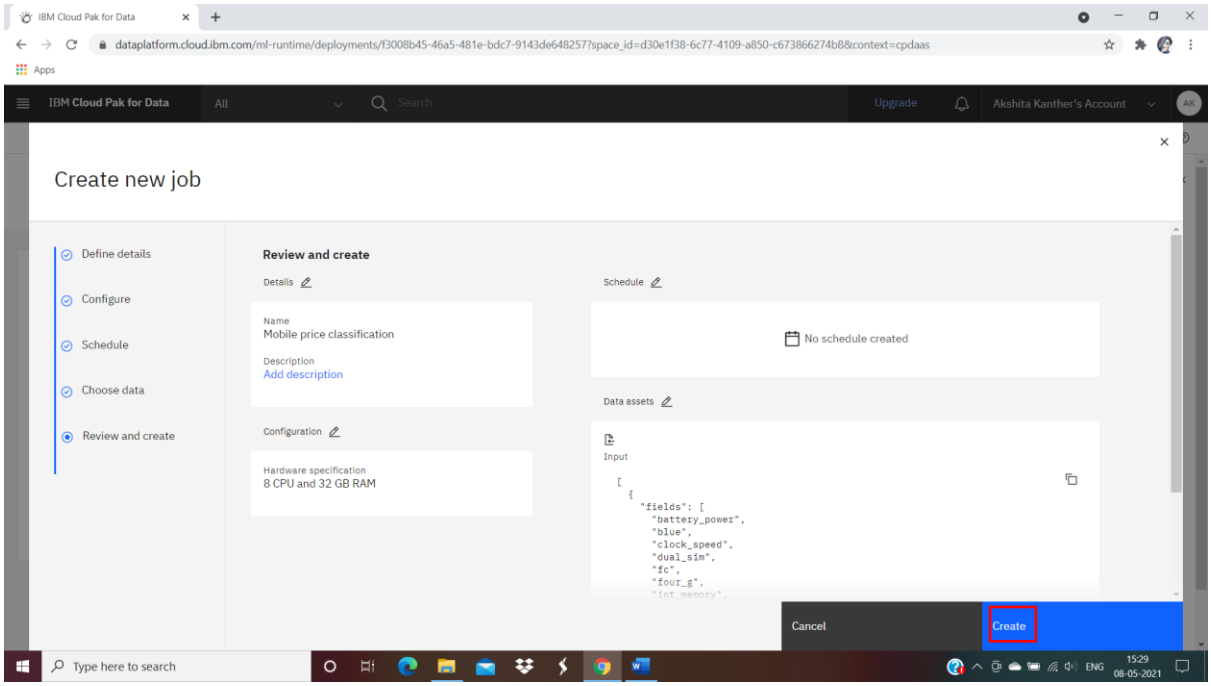
Step-55. Select Inline data for Input in Choose data.



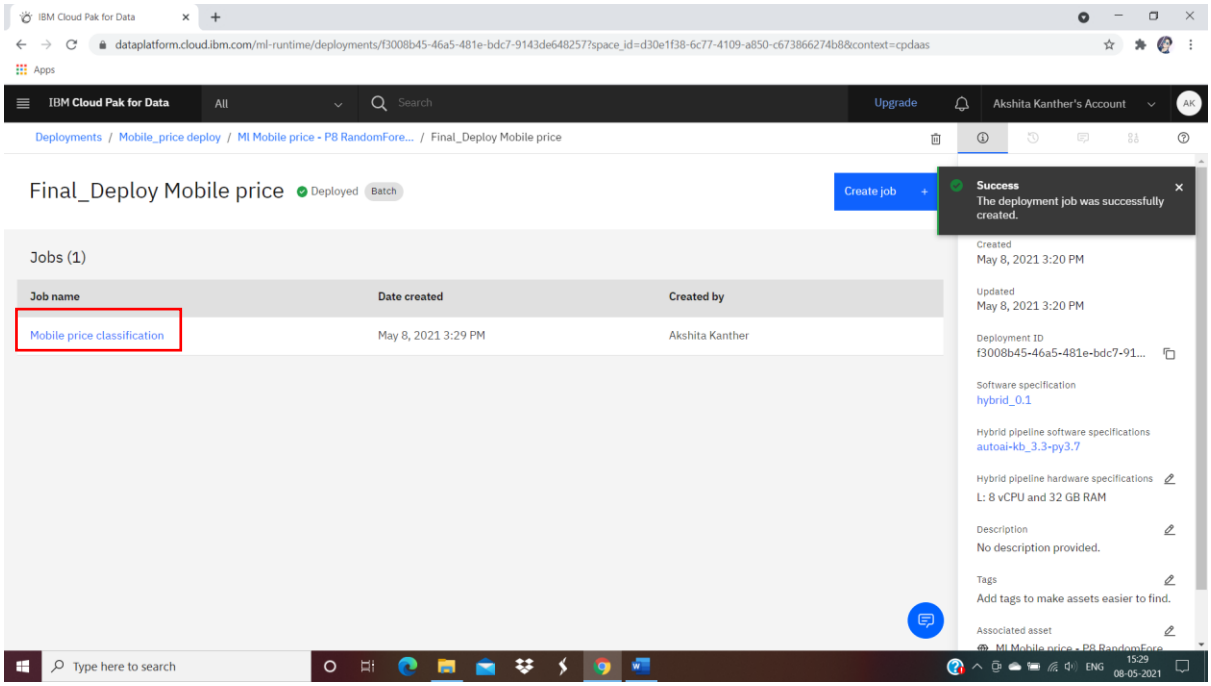
Step-56. Select Input with form and enter input values of a row for prediction and click on Next.



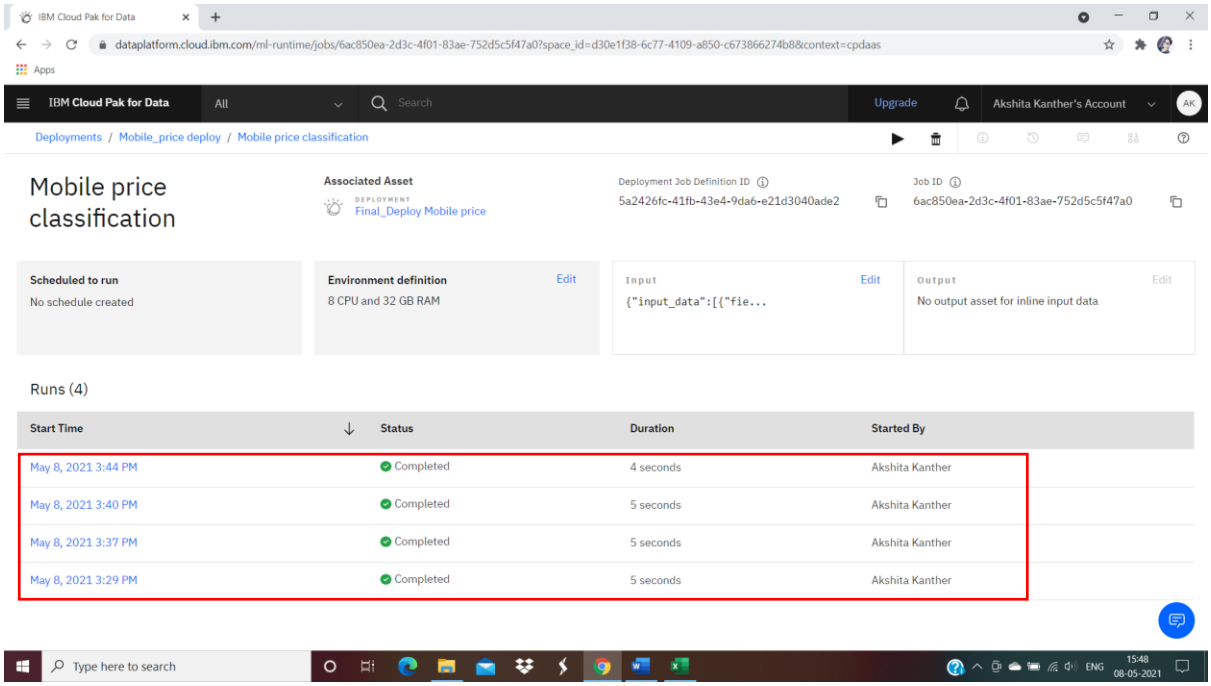
Step-57. Click on Create.

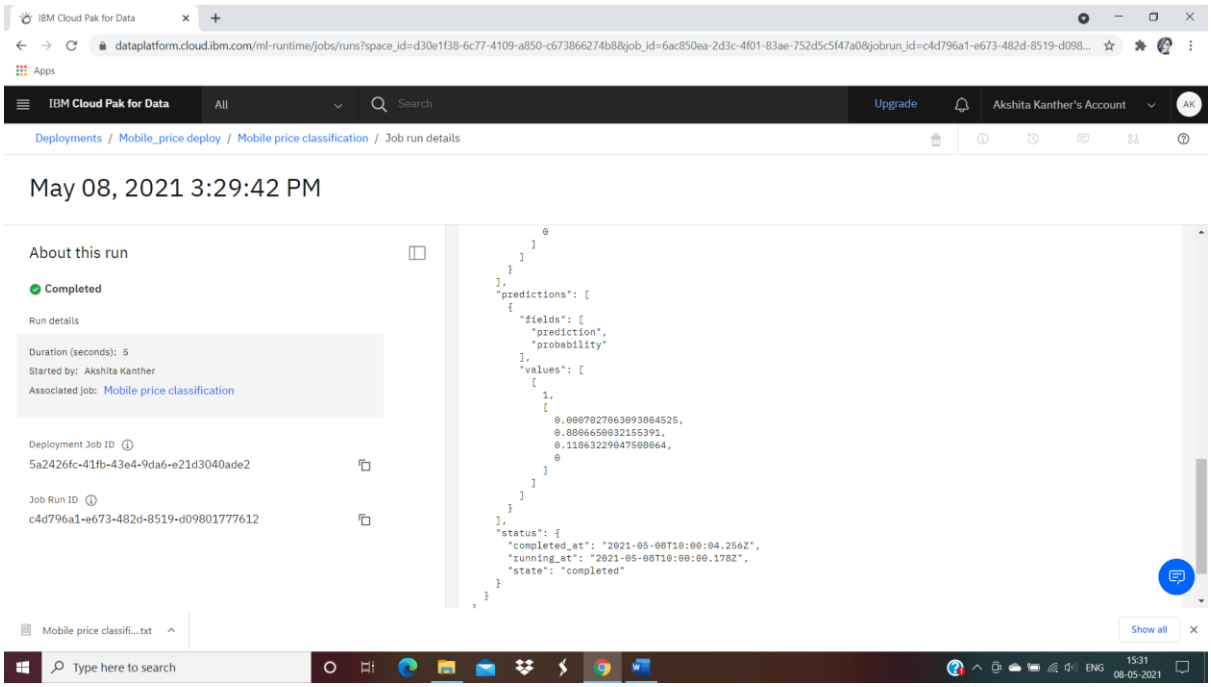


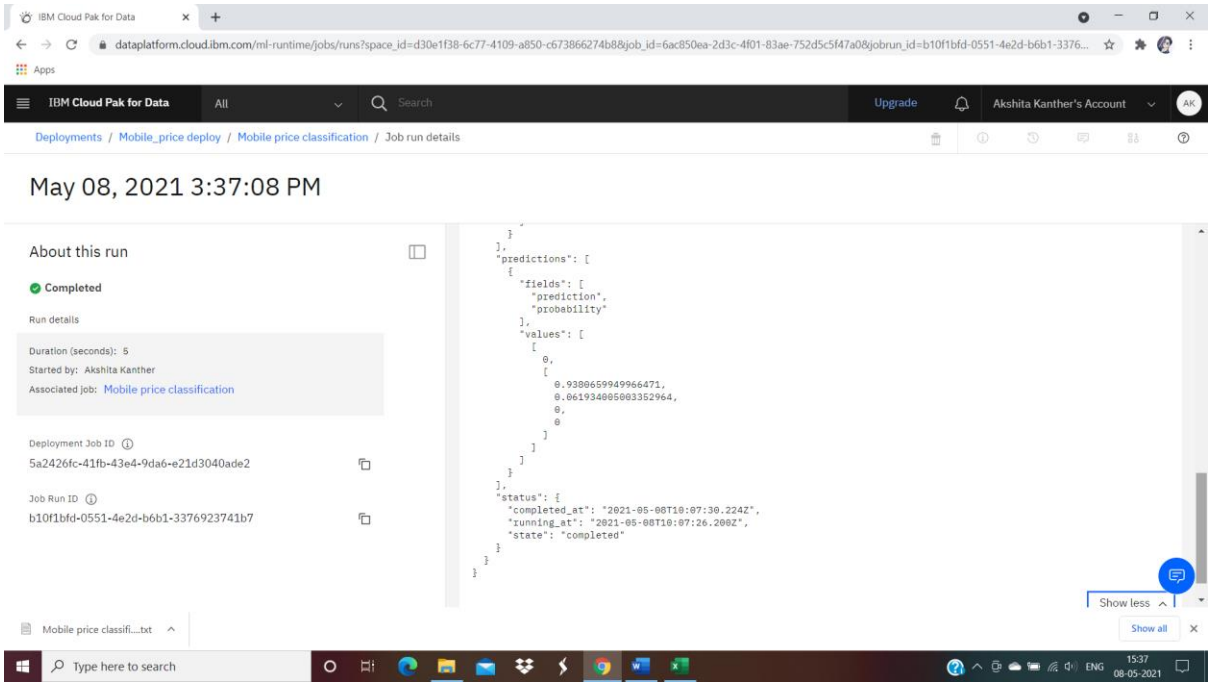
Step-58. Click on job name (Mobile price classification).

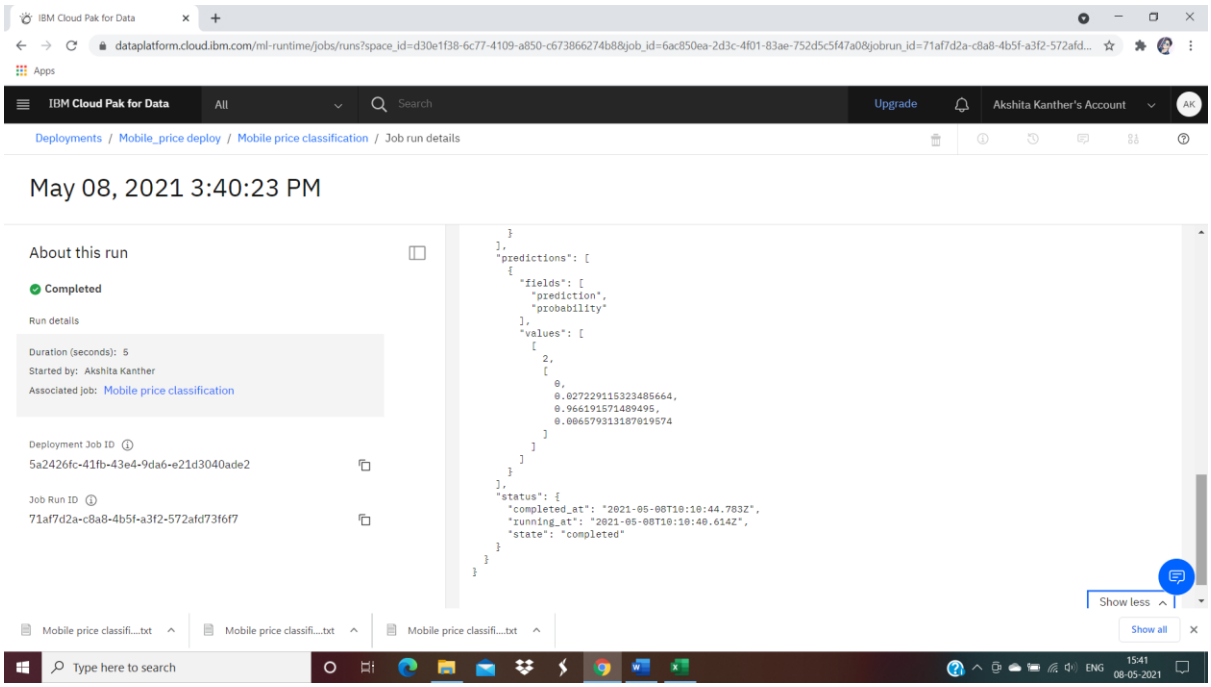


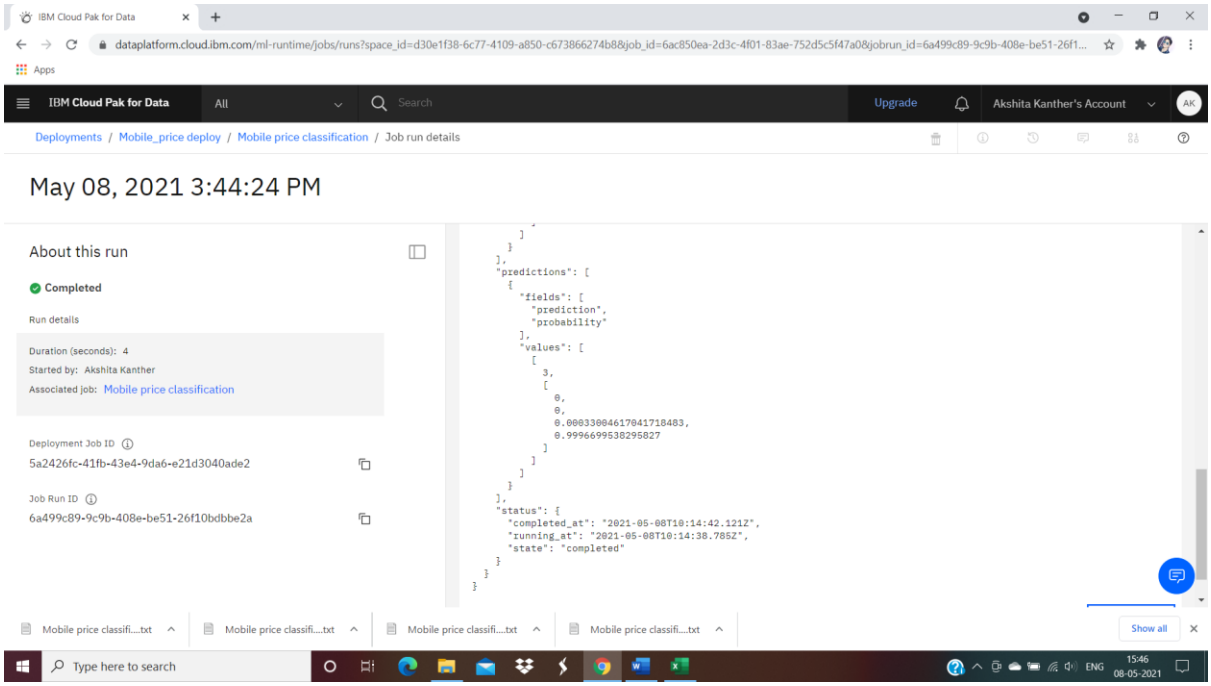
Step-59. Repeat steps 56 to 58 for different input rows from sample data. When status of all the inputs for the job is completed under Runs then click on all the links one by one under Start Time and check the output.


IBM Cloud Pak for Data
dataplatform.cloud.ibm.com/ml-runtime/jobs/6ac850ea-2d3c-4f01-83ae-752d5c5f47a0?space_id=d30e1f38-6c77-4109-a850-c673866274b8&context=cpdaas
IBM Cloud Pak for Data
All
Search
Upgrade
Akshita Kanther's Account
Deployments / Mobile_price deploy / Mobile price classification
Mobile price classification
Associated Asset
DEPLOYMENT
Final_Deploy Mobile price
Deployment Job Definition ID
5a2426fc-41fb-43e4-9da6-e21d3040ade2
Job ID
6ac850ea-2d3c-4f01-83ae-752d5c5f47a0
Scheduled to run
No schedule created
Environment definition
Edit
8 CPU and 32 GB RAM
Input
Edit
{"input_data":[{"fie...
Output
Edit
No output asset for inline input data
Runs (4)
Start Time
Status
Duration
Started By
May 8, 2021 3:44 PM
Completed
4 seconds
Akshita Kanther
May 8, 2021 3:40 PM
Completed
5 seconds
Akshita Kanther
May 8, 2021 3:37 PM
Completed
5 seconds
Akshita Kanther
May 8, 2021 3:29 PM
Completed
5 seconds
Akshita Kanther
Type here to search
ENG
15:48
08-05-2021


IBM Cloud Pak for Data
dataplatform.cloud.ibm.com/ml-runtime/jobs/runs?space_id=d30e1f38-6c77-4109-a850-c673866274b8&job_id=6ac850ea-2d3c-4f01-83ae-752d5c5f47a0&jobrun_id=c4d796a1-e673-482d-8519-d098...
IBM Cloud Pak for Data
All
Search
Upgrade
Akshita Kanther's Account
Deployments / Mobile_price deploy / Mobile price classification / Job run details
May 08, 2021 3:29:42 PM
About this run
Completed
Run details
Duration (seconds): 5
Started by: Akshita Kanther
Associated job: Mobile price classification
Deployment Job ID
5a2426fc-41fb-43e4-9da6-e21d3040ade2
Job Run ID
c4d796a1-e673-482d-8519-d09801777612
"predictions": [
"fields": [
"prediction",
"probability"
"values": [
1,
0.0007027063093804525,
0.8806650832155391,
0.11863229047508064,
0
"status": {
"completed_at": "2021-05-08T10:00:04.256Z",
"running_at": "2021-05-08T10:00:00.178Z",
"state": "completed"
Mobile price classifi....txt
Show all
Type here to search
ENG
15:31
08-05-2021


IBM Cloud Pak for Data
dataplatform.cloud.ibm.com/ml-runtime/jobs/runs?space_id=d30e1f38-6c77-4109-a850-c673866274b8&job_id=6ac850ea-2d3c-4f01-83ae-752d5c5f47a0&jobrun_id=b10f1bfd-0551-4e2d-b6b1-3376...
IBM Cloud Pak for Data
All
Search
Upgrade
Akshita Kanther's Account
Deployments / Mobile_price deploy / Mobile price classification / Job run details
May 08, 2021 3:37:08 PM
About this run
Completed
Run details
Duration (seconds): 5
Started by: Akshita Kanther
Associated job: Mobile price classification
Deployment Job ID
5a2426fc-41fb-43e4-9da6-e21d3040ade2
Job Run ID
b10f1bfd-0551-4e2d-b6b1-3376923741b7
"predictions": [
"fields": [
"prediction",
"probability"
"values": [
0,
0.9380659949966471,
0.061934005003352964,
"status": {
"completed_at": "2021-05-08T10:07:30.224Z",
"running_at": "2021-05-08T10:07:26.200Z",
"state": "completed"
Show less
Show all
Mobile price classifi....txt
Type here to search
ENG
15:37
08-05-2021

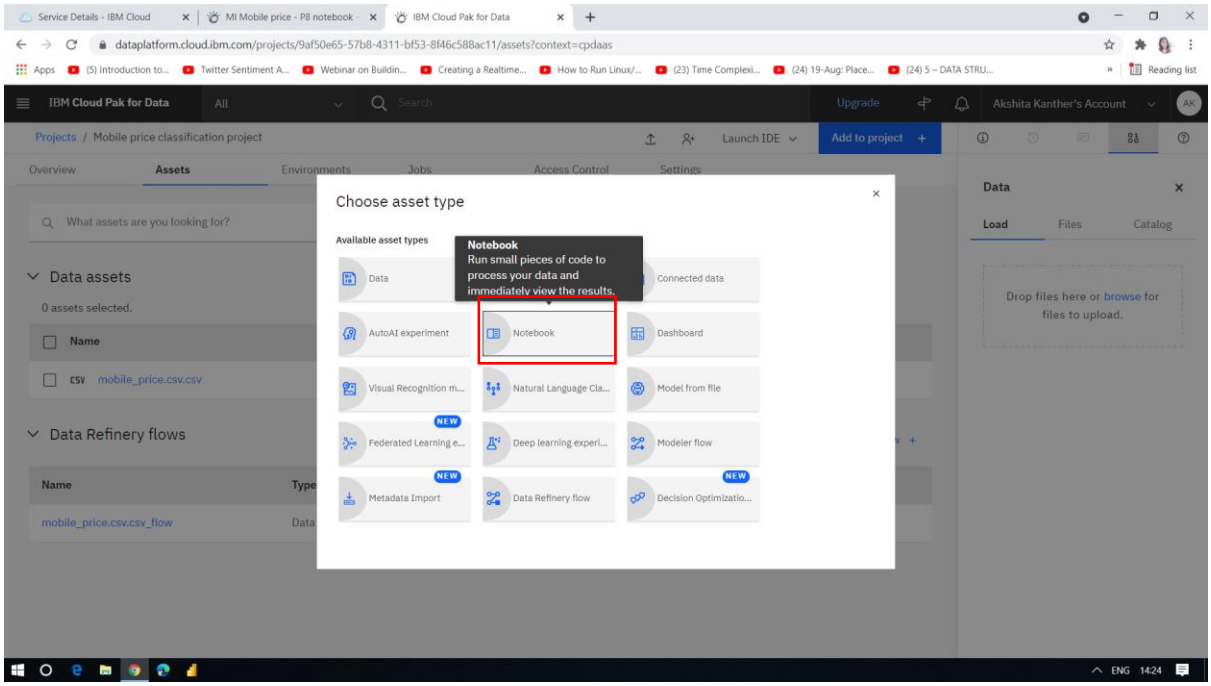

IBM Cloud Pak for Data
dataplatform.cloud.ibm.com/ml-runtime/jobs/runs?space_id=d30e1f38-6c77-4109-a850-c673866274b8&job_id=6ac850ea-2d3c-4f01-83ae-752d5c5f47a0&jobrun_id=71af7d2a-c8a8-4b5f-a3f2-572afd...
IBM Cloud Pak for Data
All
Search
Upgrade
Akshita Kanther's Account
Deployments / Mobile_price deploy / Mobile price classification / Job run details
May 08, 2021 3:40:23 PM
About this run
Completed
Run details
Duration (seconds): 5
Started by: Akshita Kanther
Associated job: Mobile price classification
Deployment Job ID
5a2426fc-41fb-43e4-9da6-e21d3040ade2
Job Run ID
71af7d2a-c8a8-4b5f-a3f2-572afd73f6f7
"predictions": [
"fields": [
"prediction",
"probability"
"values": [
2,
0,
0.027229115323485664,
0.966191571489495,
0.006579311187019574
"status": {
"completed_at": "2021-05-08T10:10:44.783Z",
"running_at": "2021-05-08T10:10:40.614Z",
"state": "completed"
Show less
Show all
Mobile price classifi....txt
Mobile price classifi....txt
Mobile price classifi....txt
Type here to search
ENG
15:41
08-05-2021



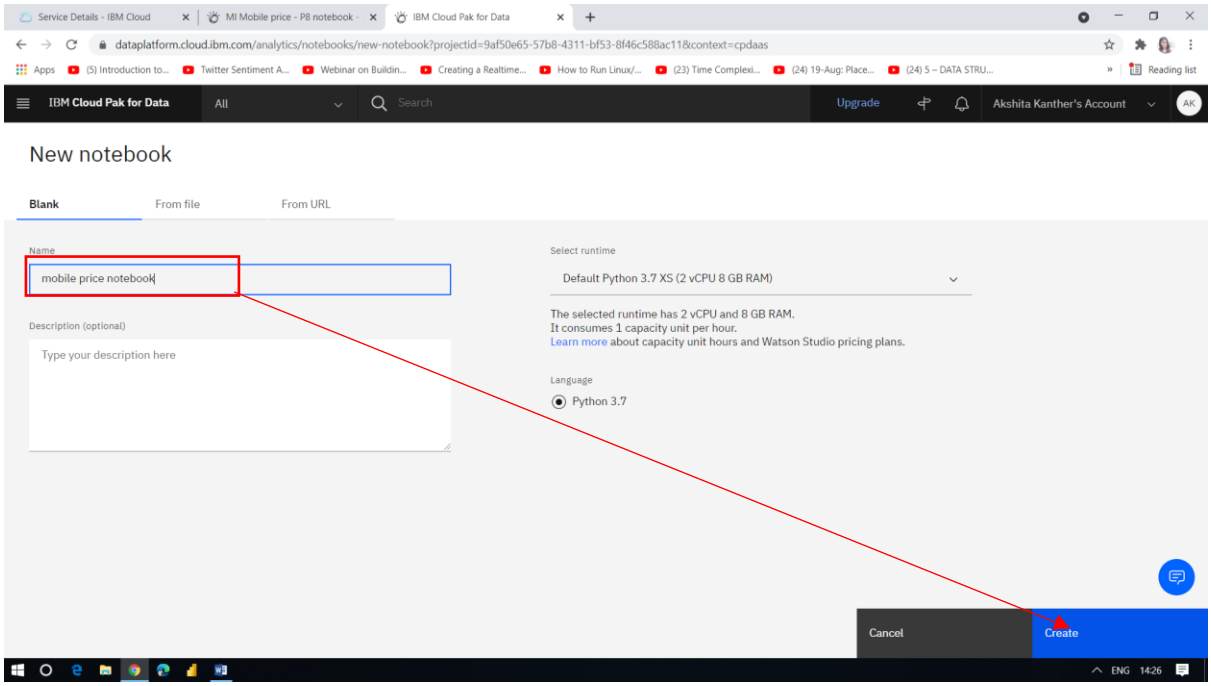
We can check predictions and probability of outputs of all the runs. After verifying it from sample data, we can conclude that since our outputs are predicted correctly with high probability for different classes, therefore our model will perform well on unknown data also.

PYTHON NOTEBOOK-

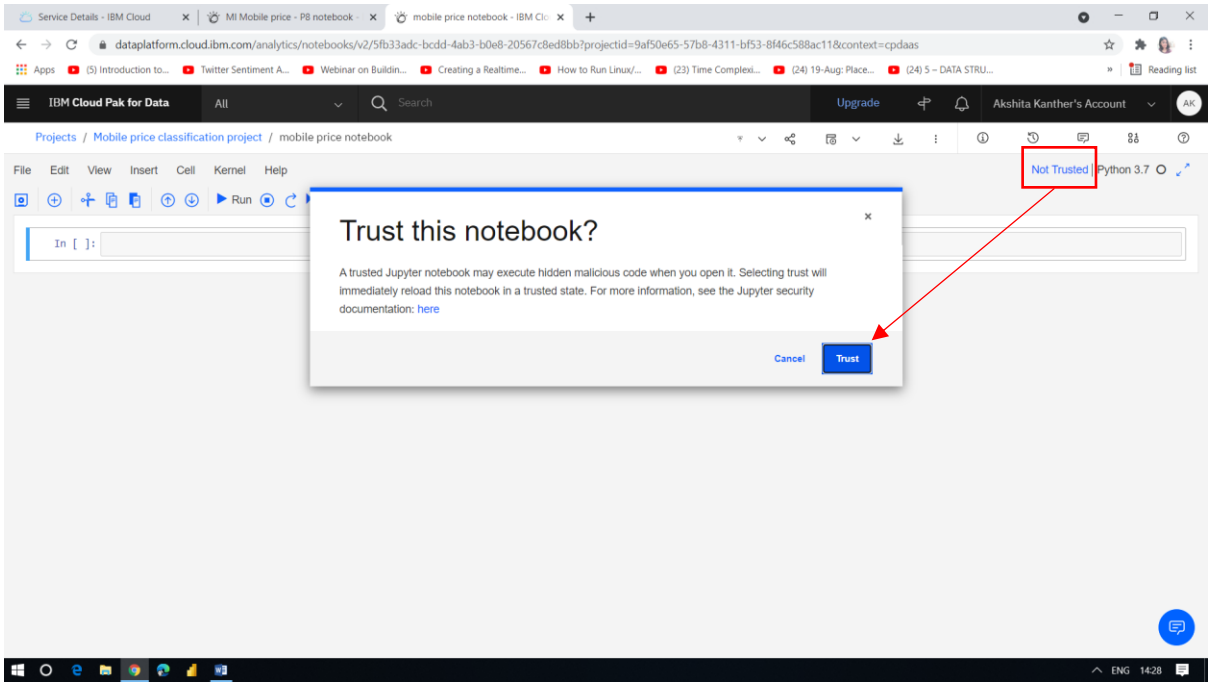
step-1. Select Add to project from Assets tab from a mobile price project. And click on Notebook.



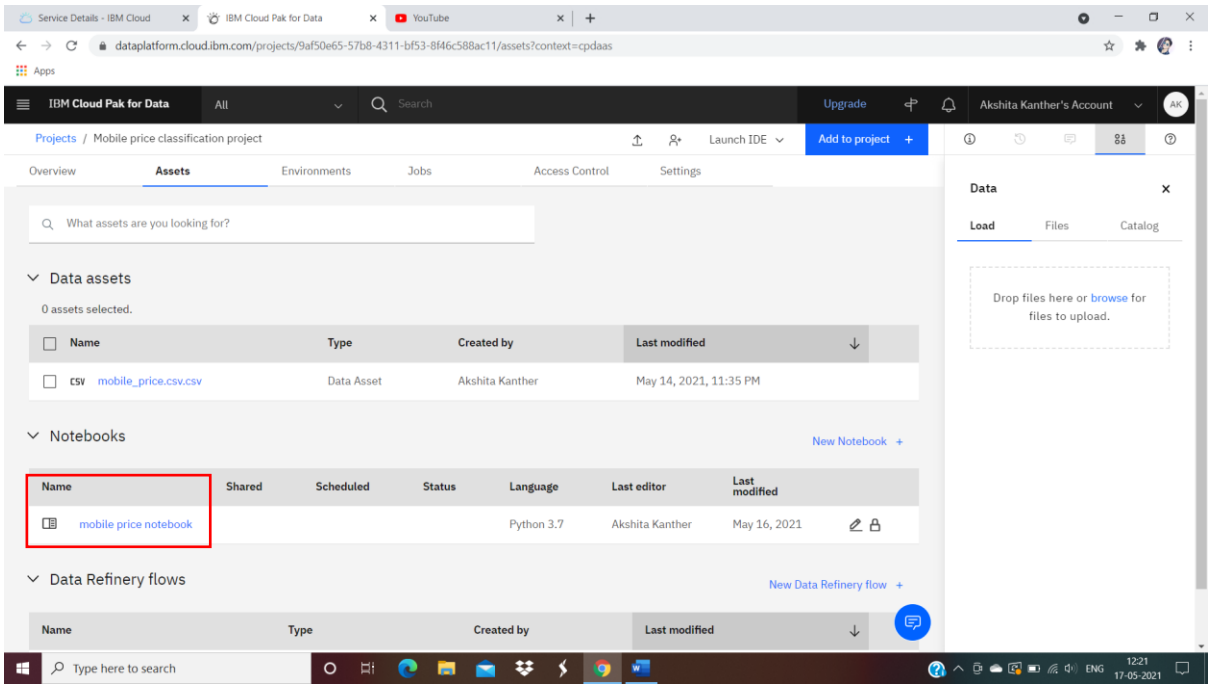
step-2. Enter a name for the notebook and click on Create.



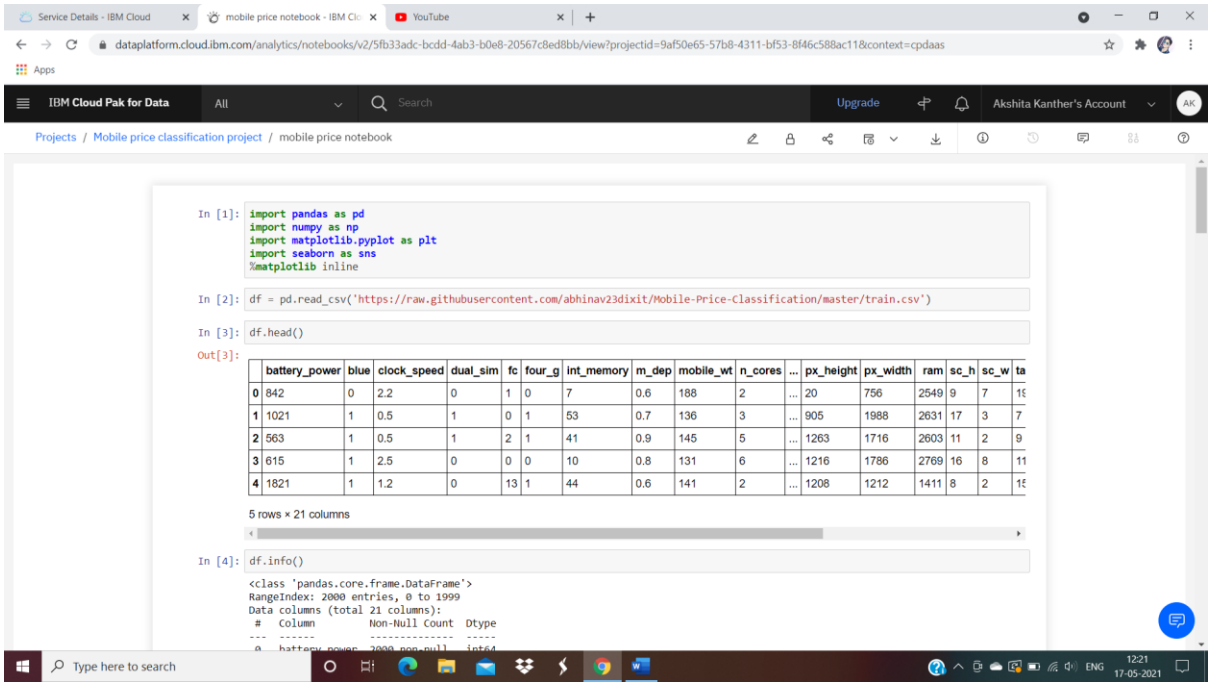
step-3. Click on Not Trusted on the top and then click on Trust.



step-4. Click on the name of the notebook under Notebooks.



step-5. Now we will write the code : firstly import the libraries , read the dataset and perform Exploratory Data Analysis.



In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

In [2]:
```
df = pd.read_csv('https://raw.githubusercontent.com/abhinav23dixit/Mobile-Price-Classification/master/train.csv')
```

In [3]:
```
df.head()
```

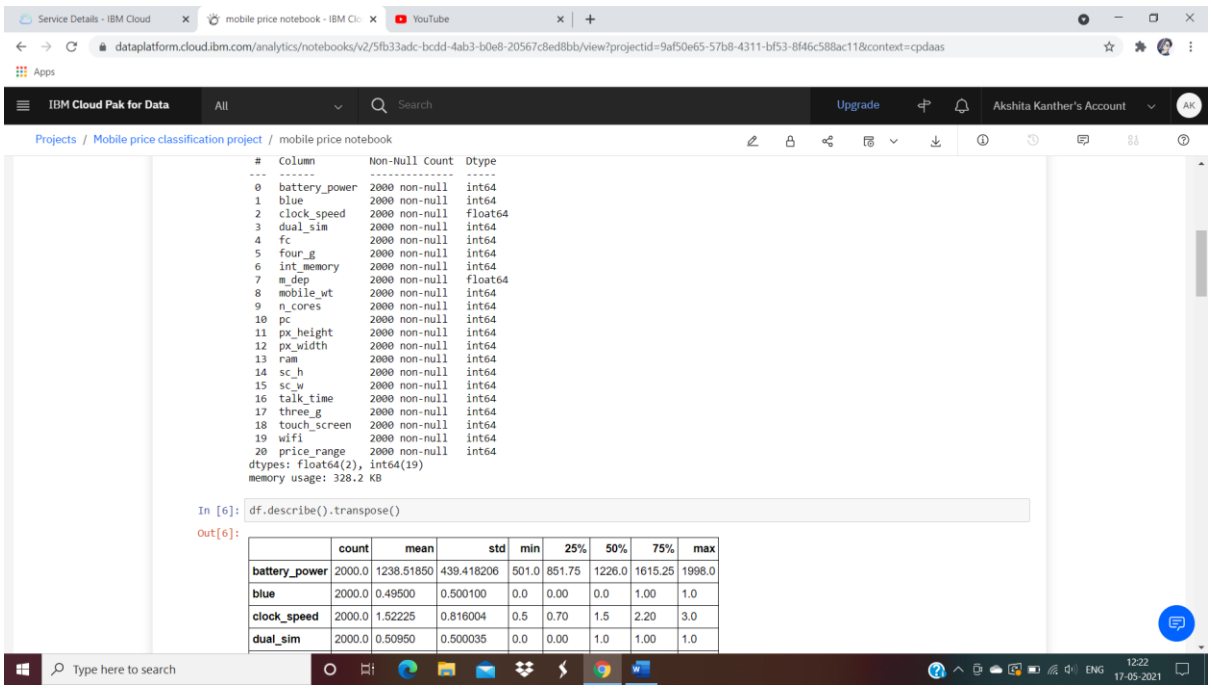
Out[3]:

	battery_power	blue	clock_speed	dual_sim	fc	four_g	int_memory	m_dep	mobile_wt	n_cores	...	px_height	px_width	ram	sc_h	sc_w	ta
0	842	0	2.2	0	1	0	7	0.6	188	2	...	20	756	2549	9	7	19
1	1021	1	0.5	1	0	1	53	0.7	136	3	...	905	1988	2631	17	3	7
2	563	1	0.5	1	2	1	41	0.9	145	5	...	1263	1716	2603	11	2	9
3	615	1	2.5	0	0	0	10	0.8	131	6	...	1216	1786	2769	16	8	11
4	1821	1	1.2	0	13	1	44	0.6	141	2	...	1208	1212	1411	8	2	15

5 rows × 21 columns

In [4]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2000 entries, 0 to 1999
Data columns (total 21 columns):
 #   Column         Non-Null Count  Dtype
```



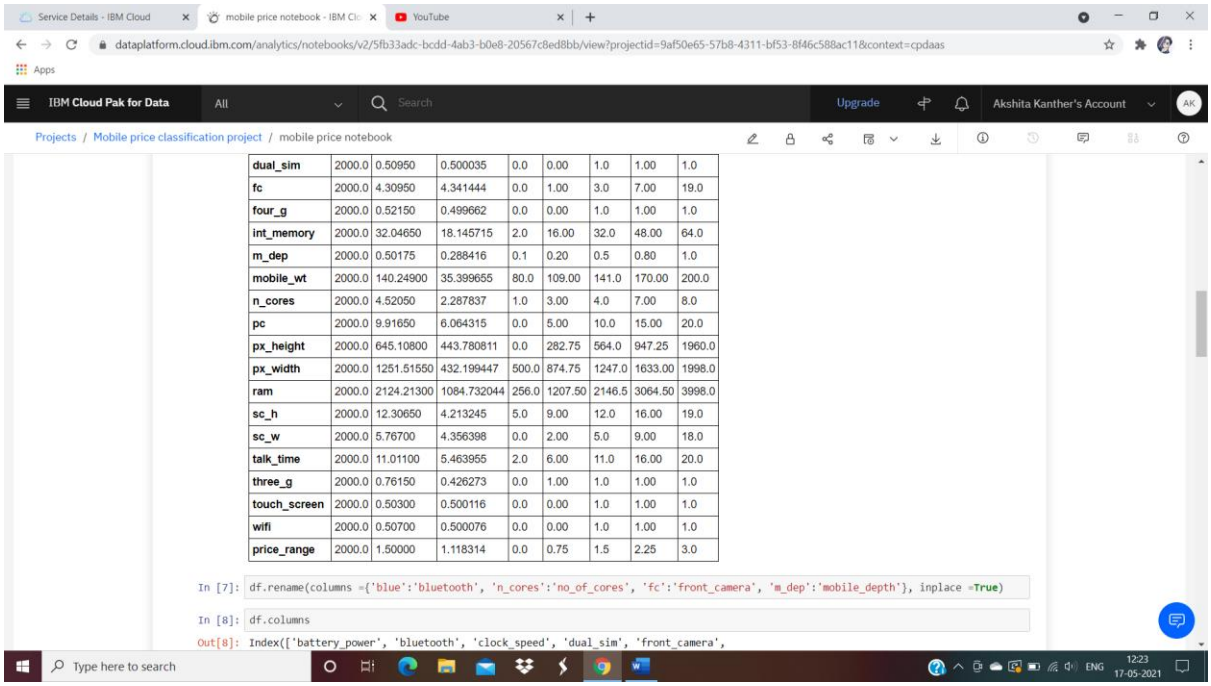
```
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   battery_power  2000 non-null   int64
 1   blue           2000 non-null   int64
 2   clock_speed    2000 non-null   float64
 3   dual_sim       2000 non-null   int64
 4   fc             2000 non-null   int64
 5   four_g         2000 non-null   int64
 6   int_memory     2000 non-null   int64
 7   m_dep          2000 non-null   float64
 8   mobile_wt      2000 non-null   int64
 9   n_cores        2000 non-null   int64
 10  pc             2000 non-null   int64
 11  px_height      2000 non-null   int64
 12  px_width       2000 non-null   int64
 13  ram            2000 non-null   int64
 14  sc_h           2000 non-null   int64
 15  sc_w           2000 non-null   int64
 16  talk_time      2000 non-null   int64
 17  three_g        2000 non-null   int64
 18  touch_screen   2000 non-null   int64
 19  wifi           2000 non-null   int64
 20  price_range    2000 non-null   int64
dtypes: float64(2), int64(19)
memory usage: 328.2 KB
```

In [6]:
```
df.describe().transpose()
```

Out[6]:

	count	mean	std	min	25%	50%	75%	max
battery_power	2000.0	1238.51850	439.418206	501.0	851.75	1226.0	1615.25	1998.0
blue	2000.0	0.49500	0.500100	0.0	0.00	0.0	1.00	1.0
clock_speed	2000.0	1.52225	0.816004	0.5	0.70	1.5	2.20	3.0
dual_sim	2000.0	0.50950	0.500035	0.0	0.00	1.0	1.00	1.0

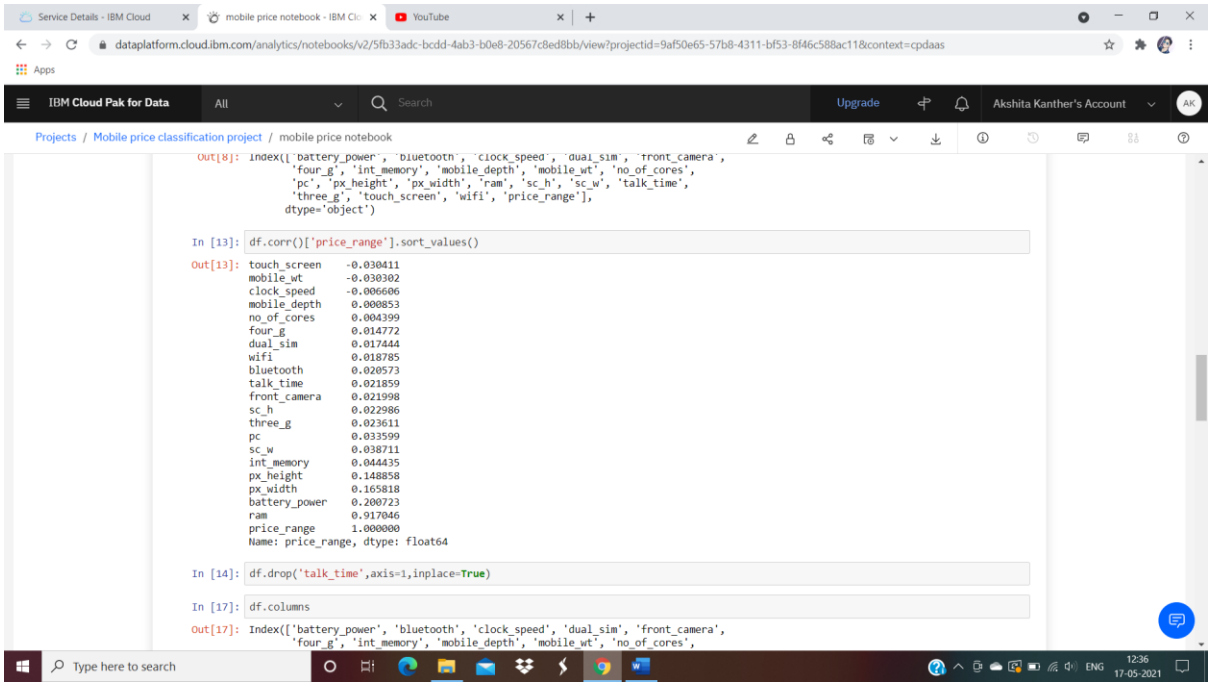
step-6. Change the column names of required columns with the rename column.



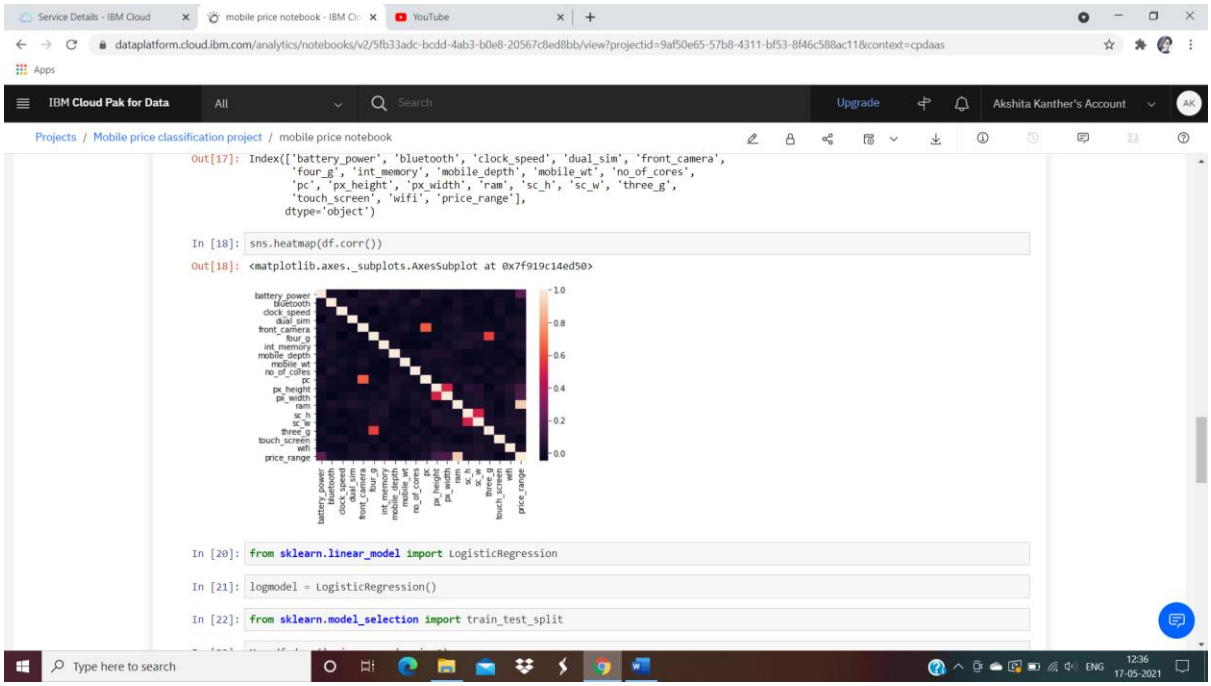
dual_sim	2000.0	0.50950	0.500035	0.0	0.00	1.0	1.00	1.0
fc	2000.0	4.30950	4.341444	0.0	1.00	3.0	7.00	19.0
four_g	2000.0	0.52150	0.499662	0.0	0.00	1.0	1.00	1.0
int_memory	2000.0	32.04650	18.145715	2.0	16.00	32.0	48.00	64.0
m_dep	2000.0	0.50175	0.288416	0.1	0.20	0.5	0.80	1.0
mobile_wt	2000.0	140.24900	35.399655	80.0	109.00	141.0	170.00	200.0
n_cores	2000.0	4.52050	2.287837	1.0	3.00	4.0	7.00	8.0
pc	2000.0	9.91650	6.064315	0.0	5.00	10.0	15.00	20.0
px_height	2000.0	645.10800	443.780811	0.0	282.75	564.0	947.25	1960.0
px_width	2000.0	1251.51550	432.199447	500.0	874.75	1247.0	1633.00	1998.0
ram	2000.0	2124.21300	1084.732044	256.0	1207.50	2146.5	3064.50	3998.0
sc_h	2000.0	12.30650	4.213245	5.0	9.00	12.0	16.00	19.0
sc_w	2000.0	5.76700	4.356398	0.0	2.00	5.0	9.00	18.0
talk_time	2000.0	11.01100	5.463955	2.0	6.00	11.0	16.00	20.0
three_g	2000.0	0.76150	0.426273	0.0	1.00	1.0	1.00	1.0
touch_screen	2000.0	0.50300	0.500116	0.0	0.00	1.0	1.00	1.0
wifi	2000.0	0.50700	0.500076	0.0	0.00	1.0	1.00	1.0
price_range	2000.0	1.50000	1.118314	0.0	0.75	1.5	2.25	3.0

```
In [7]: df.rename(columns ={'blue':'bluetooth', 'n_cores':'no_of_cores', 'fc':'front_camera', 'm_dep':'mobile_depth'}, inplace =True)

In [8]: df.columns
Out[8]: Index(['battery_power', 'bluetooth', 'clock_speed', 'dual_sim', 'front_camera',
```

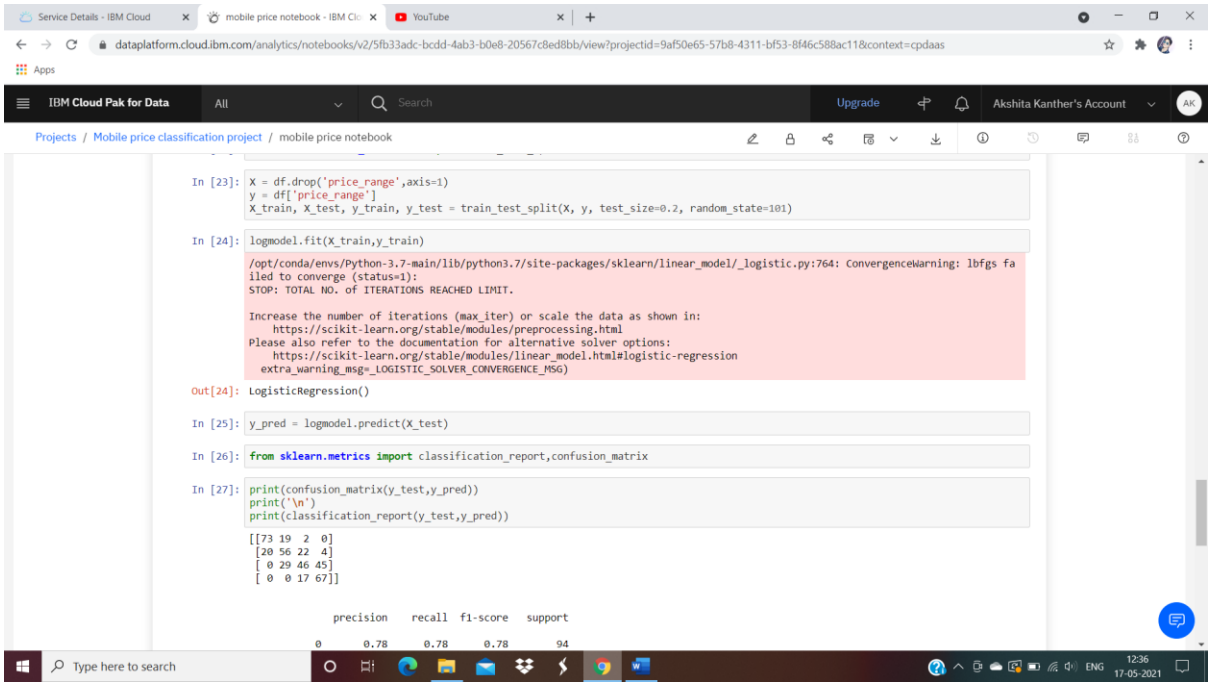
step-7. We will check the correlation of all the columns with the price_range column and sort it in ascending order. We can observe that the ram column is highly correlated with the price_range column. We will drop the columns which are not required for analysis.

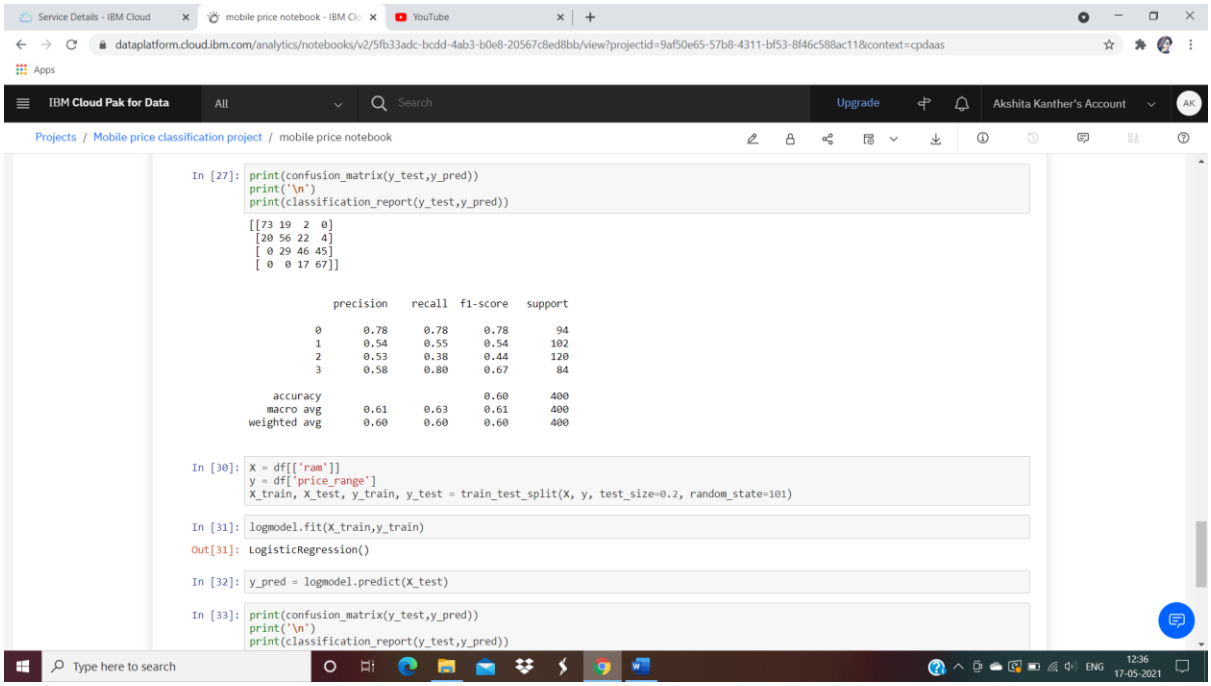


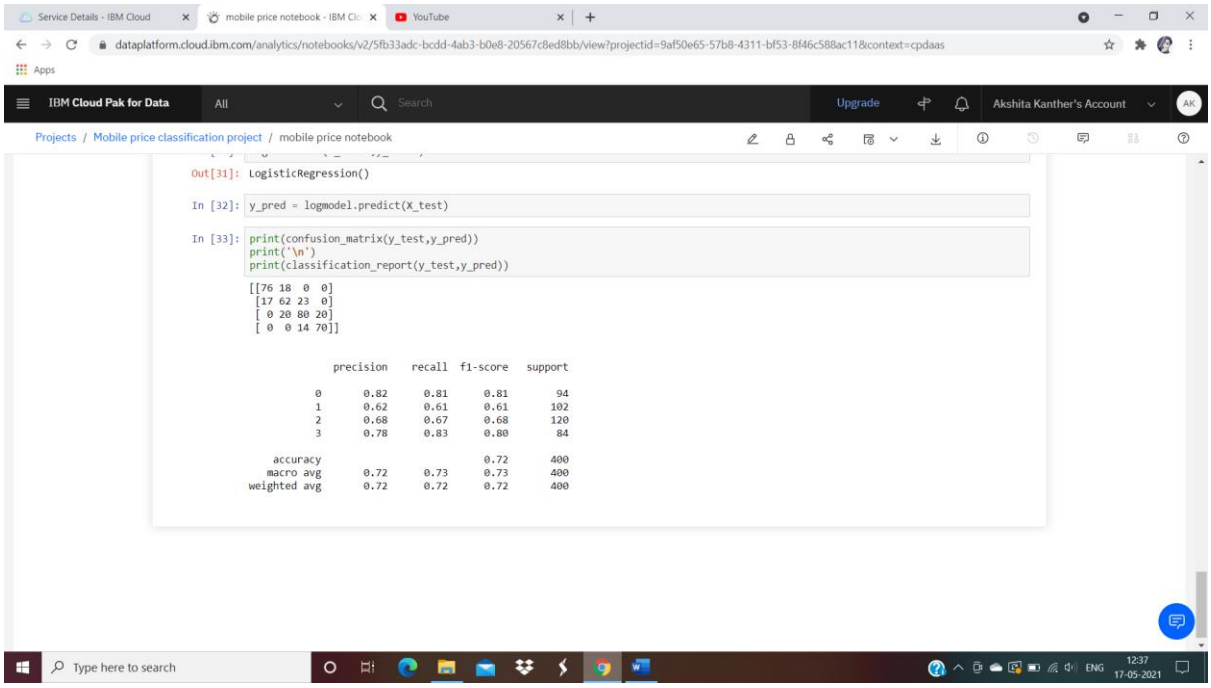
```
Out[8]: Index(['battery_power', 'bluetooth', 'clock_speed', 'dual_sim', 'front_camera',
               'four_g', 'int_memory', 'mobile_depth', 'mobile_wt', 'no_of_cores',
               'pc', 'px_height', 'px_width', 'ram', 'sc_h', 'sc_w', 'talk_time',
               'three_g', 'touch_screen', 'wifi', 'price_range'],
              dtype='object')

In [13]: df.corr()['price_range'].sort_values()

Out[13]: touch_screen    -0.030411
         mobile_wt       -0.030302
         clock_speed     -0.006606
         mobile_depth     0.000853
         no_of_cores      0.004399
         four_g           0.014772
         dual_sim         0.017444
         wifi             0.018785
         bluetooth        0.020573
         talk_time        0.021859
         front_camera     0.021998
         sc_h             0.022986
         three_g          0.023611
         pc               0.033599
         sc_w             0.038711
         int_memory       0.044435
         px_height        0.148858
         px_width         0.165818
         battery_power    0.200723
         ram              0.917046
         price_range      1.000000
         Name: price_range, dtype: float64

In [14]: df.drop('talk_time',axis=1,inplace=True)

In [17]: df.columns
Out[17]: Index(['battery_power', 'bluetooth', 'clock_speed', 'dual_sim', 'front_camera',
                'four_g', 'int_memory', 'mobile_depth', 'mobile_wt', 'no_of_cores',
```



step-8. Now split the data , train and evaluate the model. We will create a logistic regression model as we want to predict the output which will be in different classes (0,1,2,3 : low price, medium price ,high price, very high price).







We have created two models. In the first model we have predicted output on the basis of all the columns except price_range column ,the accuracy and F1-score is not very good because price_range is weakly correlated with all other columns except ram , in the second model, we have predicted price range on the basis on only ram column because they were highly correlated. And we have observed significant increase in the F1-score and accuracy of the model.

CONCLUSION

Watson is IBM's portfolio of business-ready tools, applications and solutions, designed to reduce the costs and hurdles of AI adoption while optimizing outcomes and responsible use of AI. It operationalizes AI and transforms how work gets done with our proven capabilities and experiences .

After conducting the above project, we learned about how to work with the IBM Data Refinery tools, Visual Recognition service, Machine Learning model(Auto AI experiment) and Python Notebook on IBM cloud platform using IBM Watson Service . We were able to train the model for our dataset according to the requirement of the customer. We came to know how to work on a data set, how to classify the images and train the model according to classes needed and show results according to that classes. We came across different machine learning algorithms, we implemented Logistic Regression algorithm in Python Notebook as we had to predict the output in the form of classes/ categories. In IBM Auto AI experiment different pipelines were created , we evaluated different models on the basis of F1-score , Accuracy, Precision, Recall and various other model evaluation measures.