Woocation Internship

CLASSIC MACHINE LEARNING

DETECT NEXT HOLIDAY LOCATION - GIVEN PARAMETERS LIKE WEATHER, PRICE, FLIGHT TIME, HOTEL PRICE AND OTHER FEATURES DEVELOP MACHINE LEARNING SYSTEM WHICH LEARNS AS USER CLICKS ON LOCATION AND SUGGEST THEM NEXT HOLIDAY LOCATION

Introduction

Recommendation System are needed everywhere. Whether you are looking for a movie you might enjoy watching or a book that you might enjoy reading.

For the given task, I have used the **Azure Machine Learning Platform.** Azure Machine Learning Studio provides the tools you need to develop a recommendation model and then operationalize it by deploying it as an Azure web service.

How Does this Recommender Systems Work?

Typically, there are two types of entities involved in a recommender system let's call them *users* and *items*. Users are the people to whom you would like to make recommendations. Items are the things you would like to recommend to them such as movies, books and **Holiday location** as in the given case.

In the given case, we would like to recommend, a holiday location to a given user based on the user clicks that this user and other users have provided for some of the Holiday Location and parameters like weather, price, flight time, hotel price.

How to Predict Next Holiday Location?

In order to build an Machine Learning model that can predict, for a given user/item combination, For the given case, we need to collect data of the form (UserID, Location, Clicks).

- UserId-The Id of the User
- Location-The location the user is looking for
- Clicks-The number of clicks the user makes on the Location

I created a Random dataset of the given type which looks like this :-
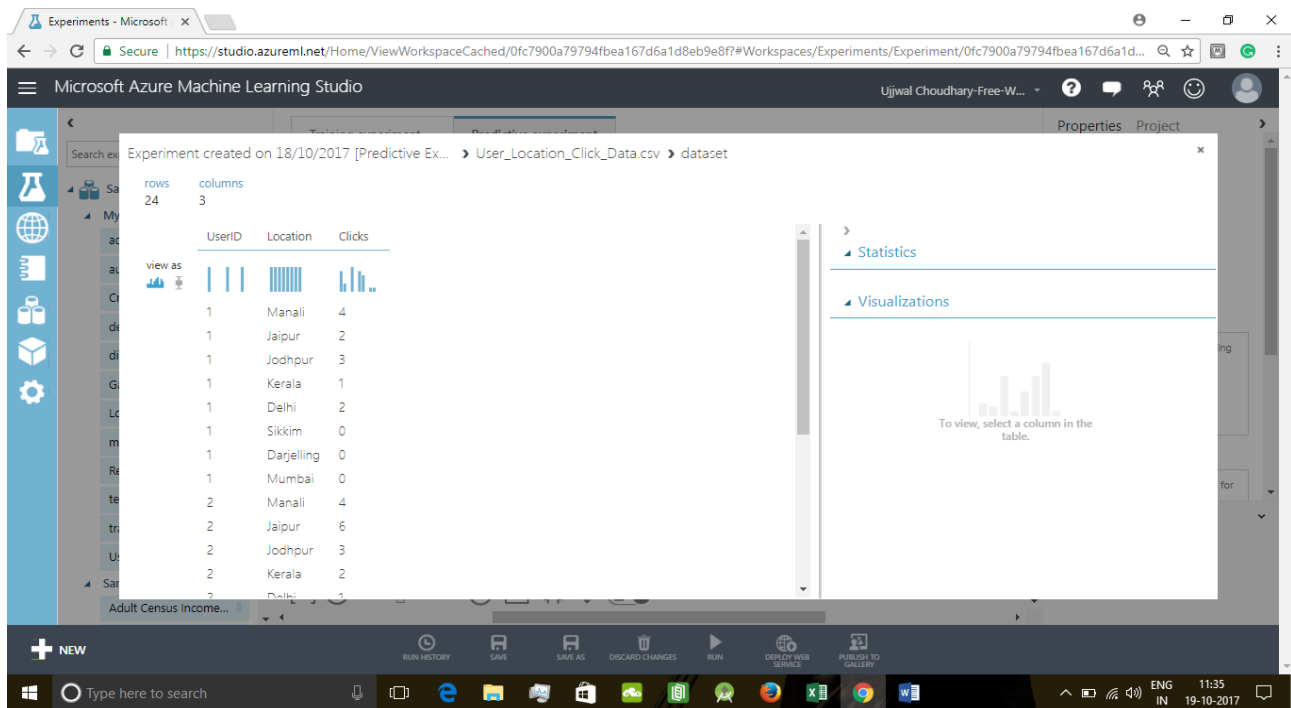
- This will be a sparse matrix because typical users will only click a small subset of items. The Bayesian RS implemented in Azure ML takes this training data, trains a model, and essentially returns a function that predicts for a given user/item pair how the user would like that item.
- To train data with the given parameters of the location. I created a csv file containing attributes of the location such as Minimum Hotel Price, Airport, Weather/Temperature.

The file looks like this :-
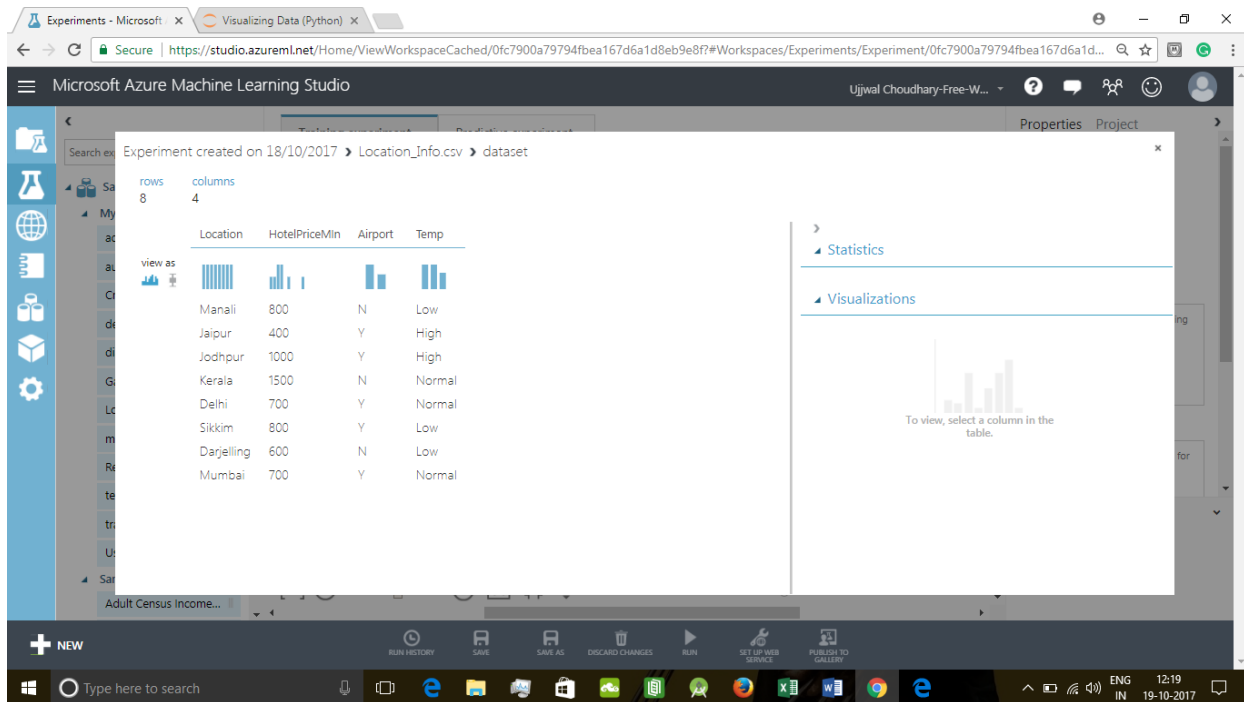
To split data into training and validation set, I have added the **split data module**. Because this a recommender system, I have used the recommender split data as the desired splitting mode so that the number of clicks of the same user and location get grouped correctly.
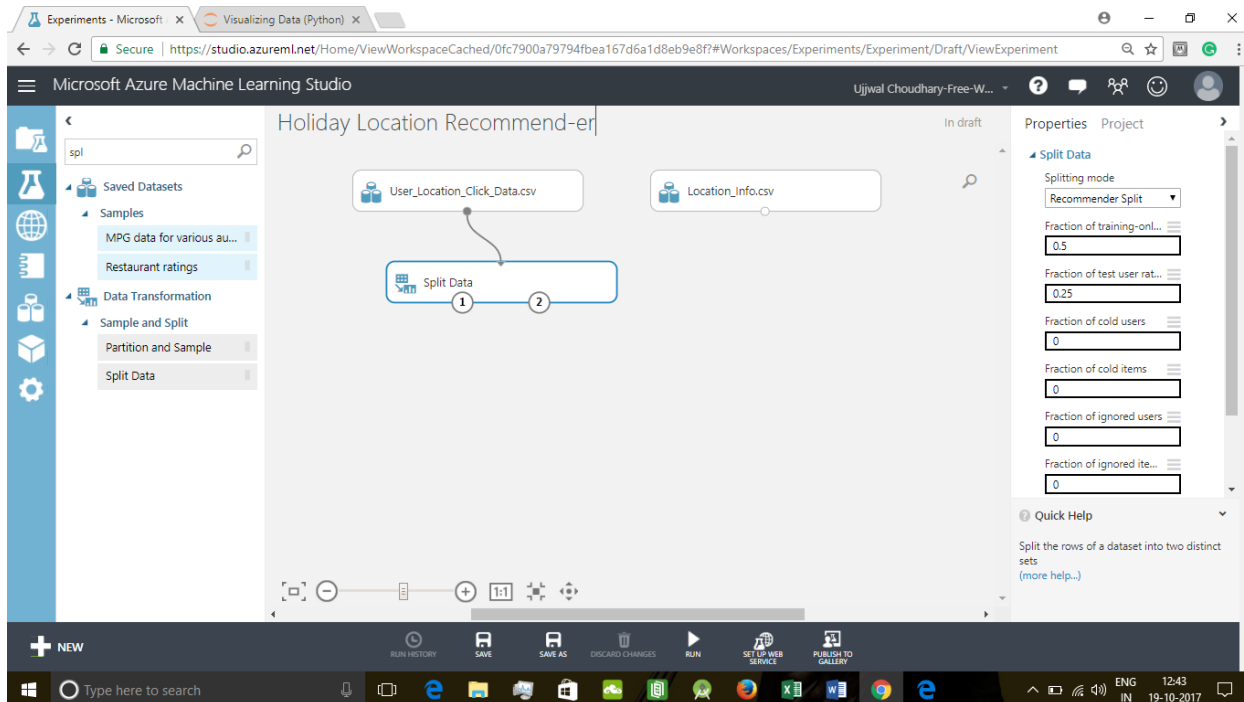
We then use the **Train Matchbox Recommender** module to train the model. **MatchBox is the name of the algorithm.**

The left output of the split data is the training data and will be the operational input for the Train Matchbox Recommender module. The second input will be for the User Feature, the parameters of the user or the user data. In the given case, we don't have any user data so it is left blank. The third input will be the item features, In this case the parameters of the location or the location data
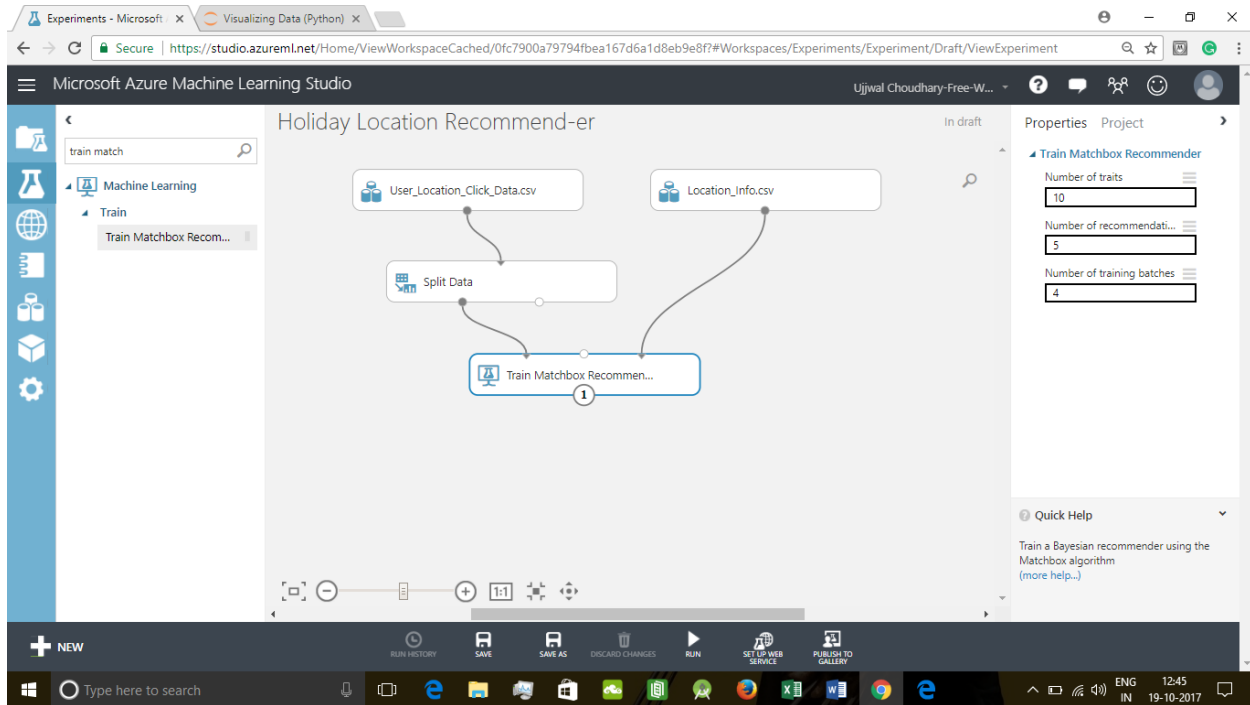
To validate the model, we scored against the validation data. I have used the **Score matchbox recommende**r. The output of the Train Matchbox Recommender act as the input of the Score Model.

The second input is for the validation data which comes from the right output of the spilt data module

Here again the third input is for the user features and forth for the item features. Since we don't have any user data the second input is left blank.

The Model has four different prediction options for different recommendation systems. I have used Item Recommendation which in our case means given a user predict the list of holiday location.
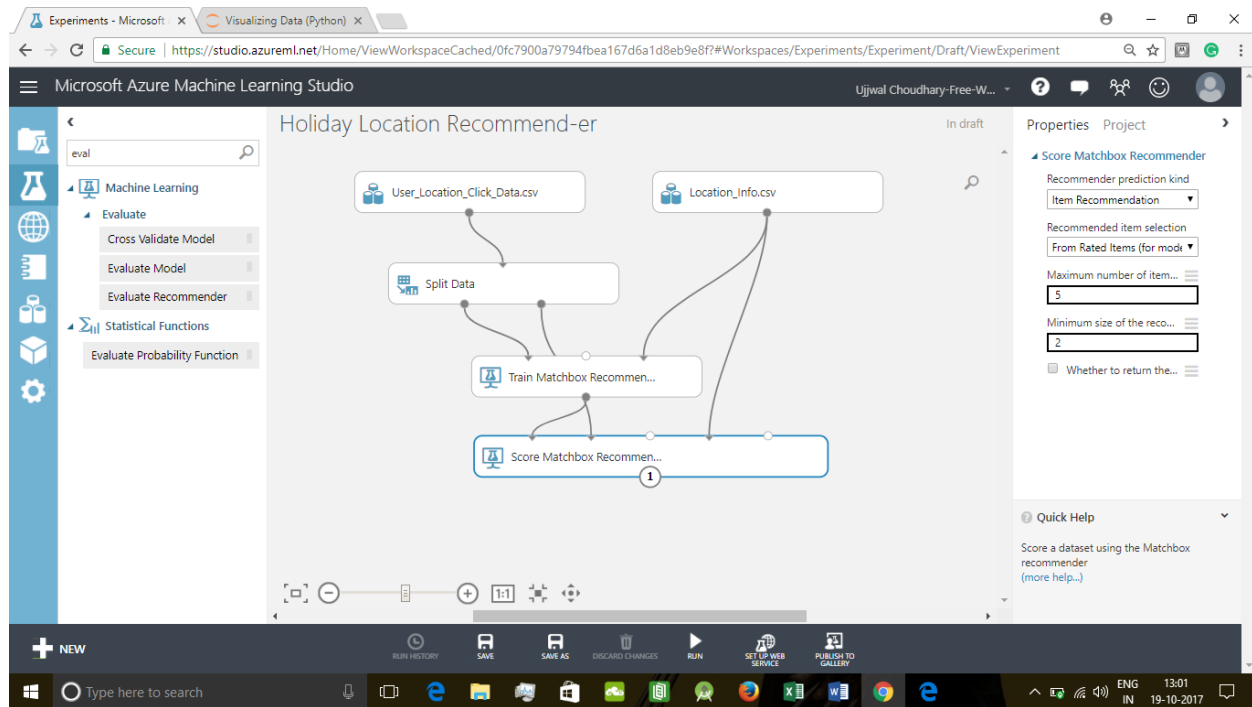
To evaluate the model. I have added the **Evaluate recommender model.** The output of the score model act as the right input of evaluate Recommender Model. The Validation data act as the left input of the evaluation model.
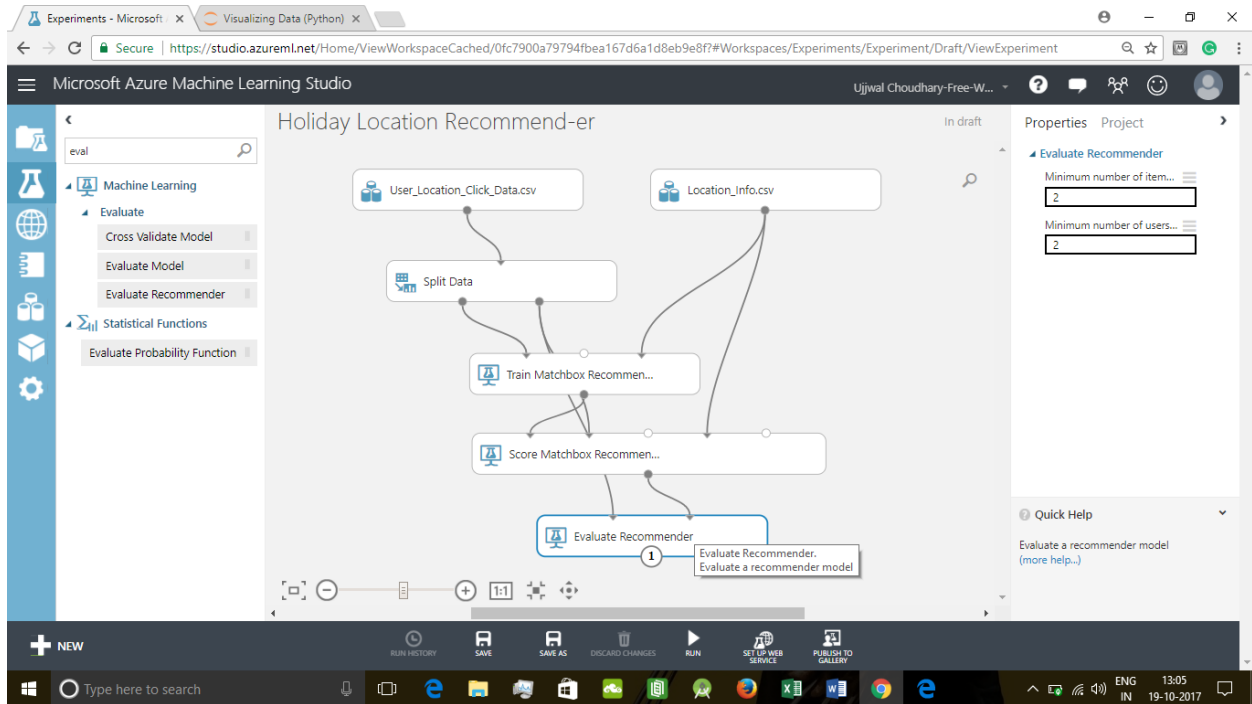
After running the experiment, we take a look at the results.

The Scoring output shows a set of recommended holiday location by UserID.

One row for each User
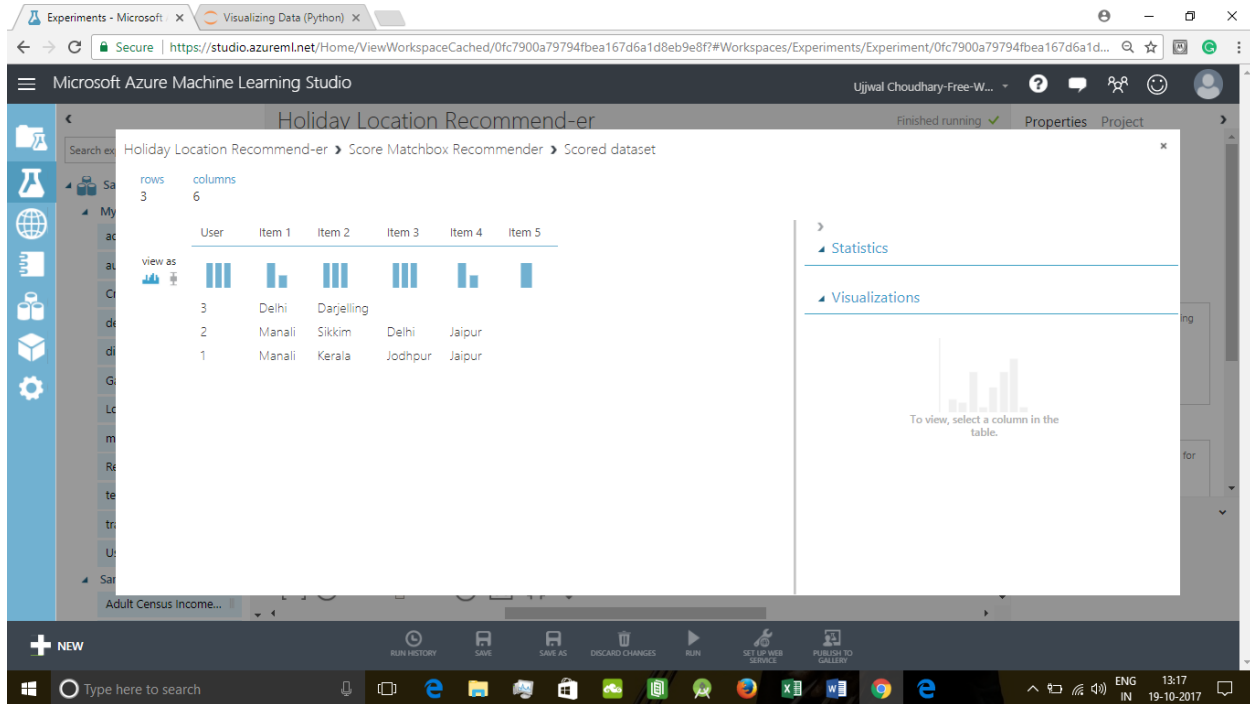
The evaluation shows a metric called normalized discounted cumulative gain(NDCG) which measures how the recommendation match the actual preferences on a scale from zero to one.

In our model, it is 0.9378 which means the model is 93.78% accurate.
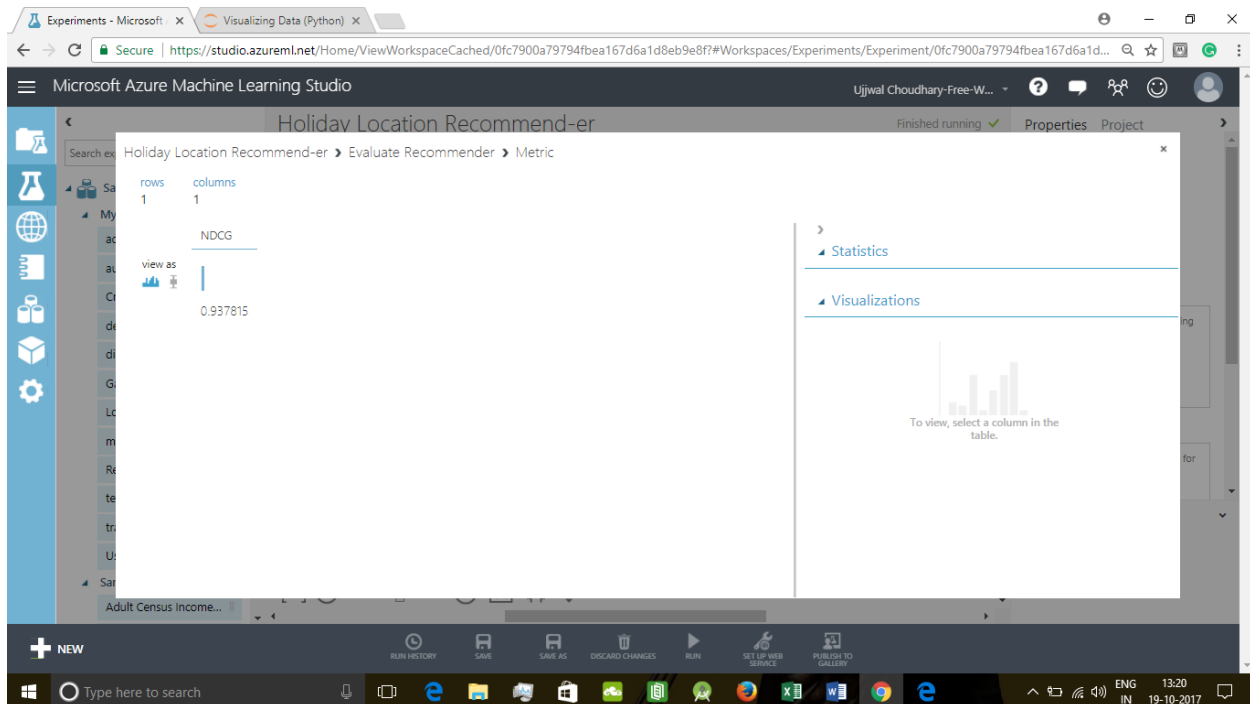
Now let's put this model into production as a web-service
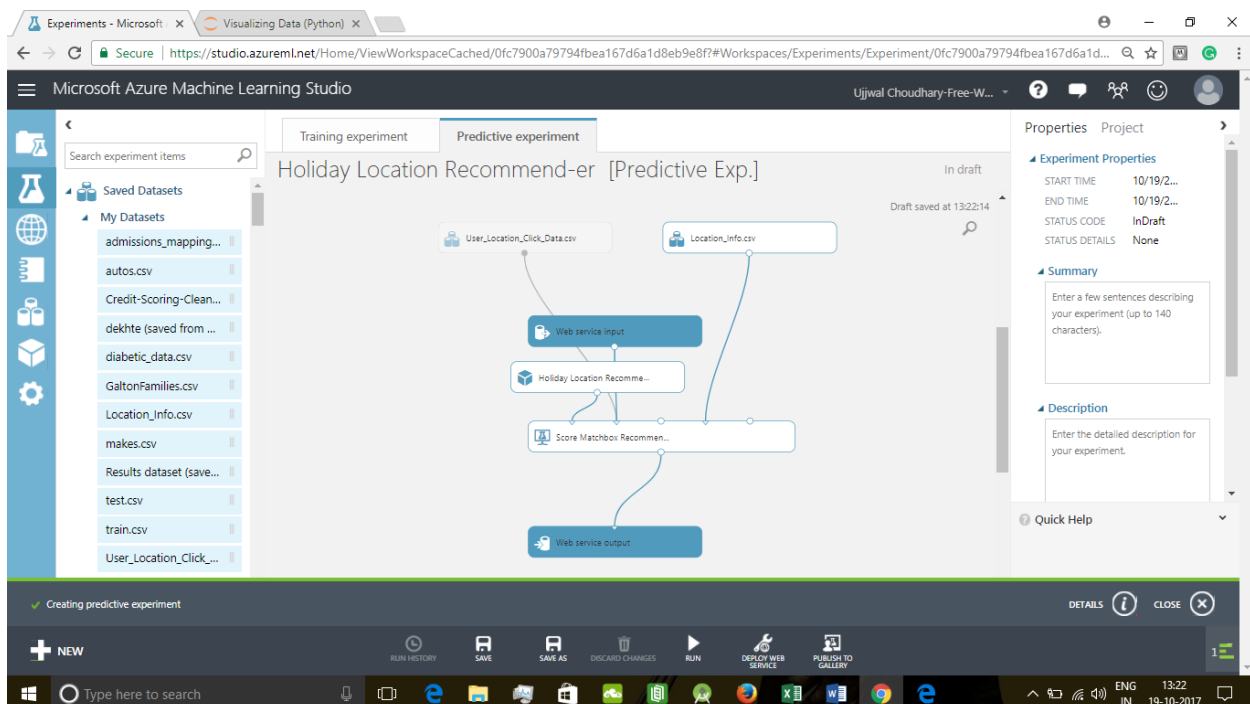
I added the Select Column in the database module to include only the UserID as the input.
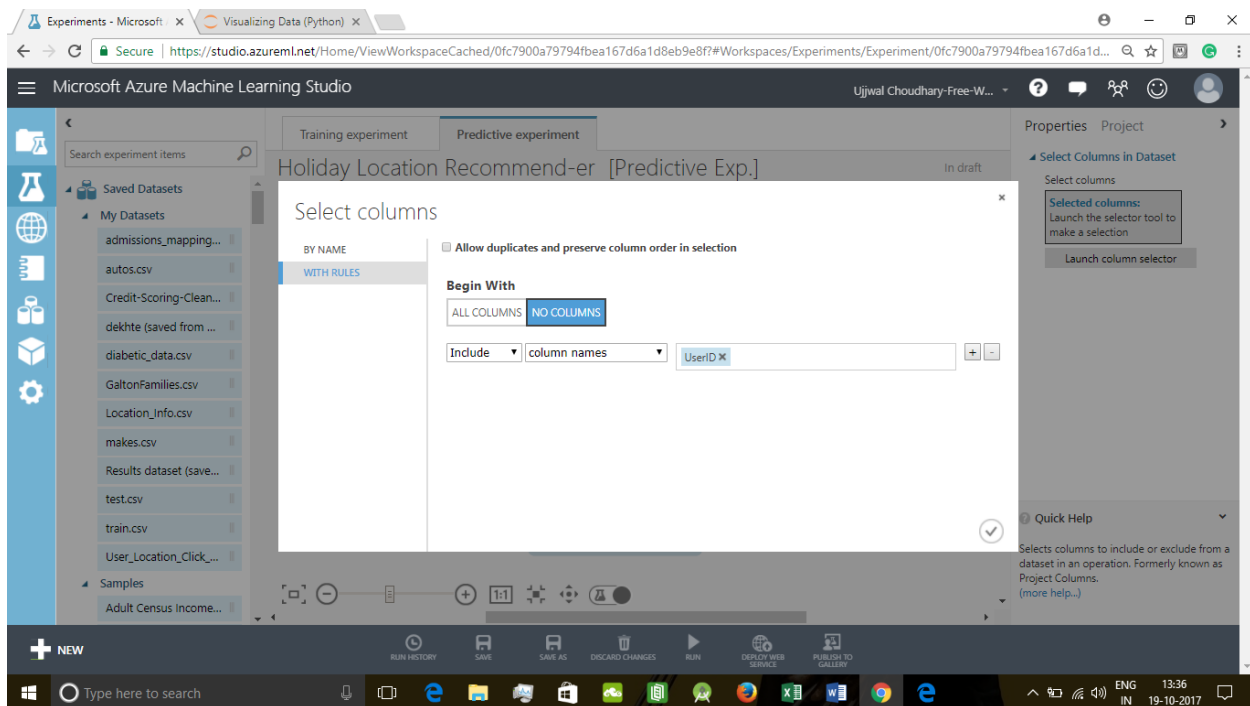
This is now how the predictive experiment looks:
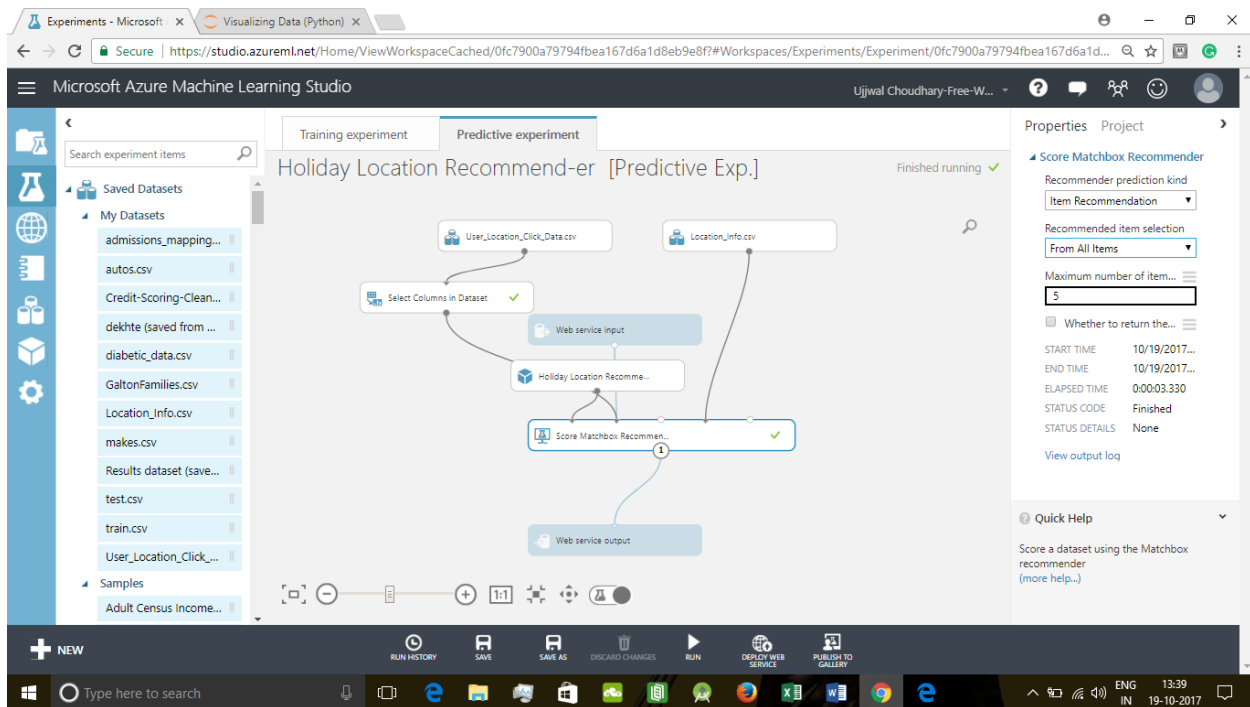
Now we are going to deploy it as a web service and Check the Result

RESULTS
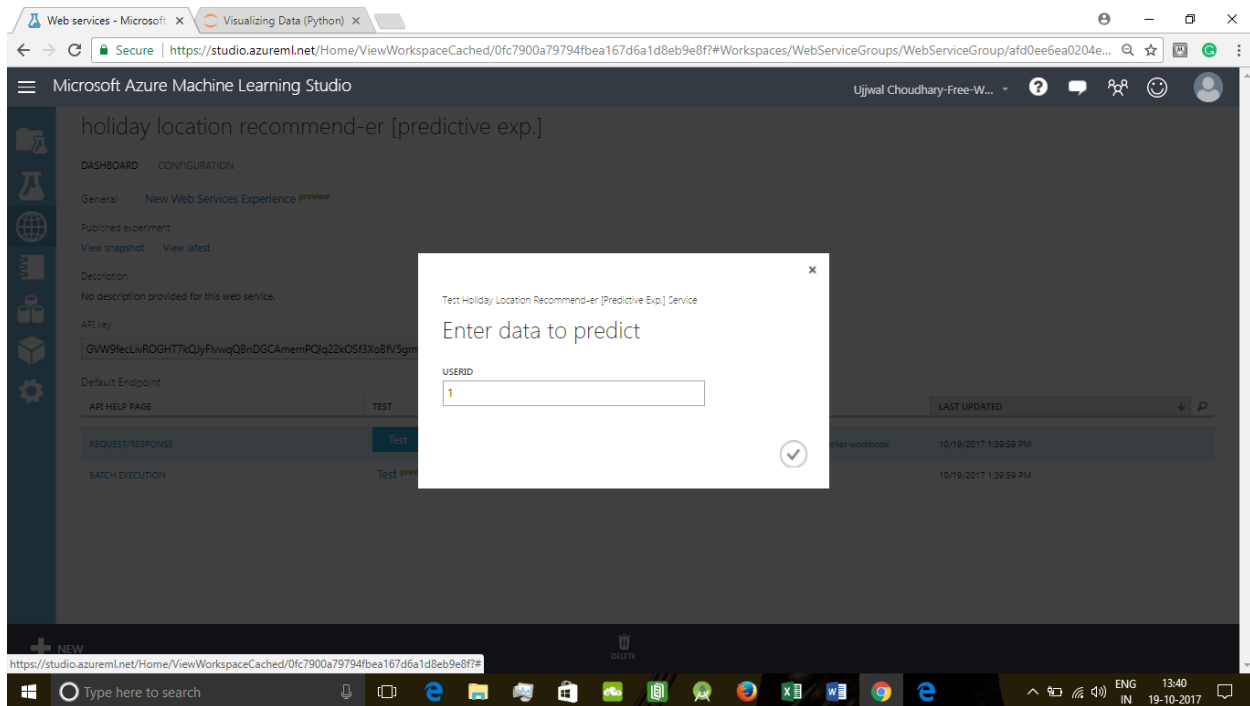
The system recommended the following locations for UserID 1:-
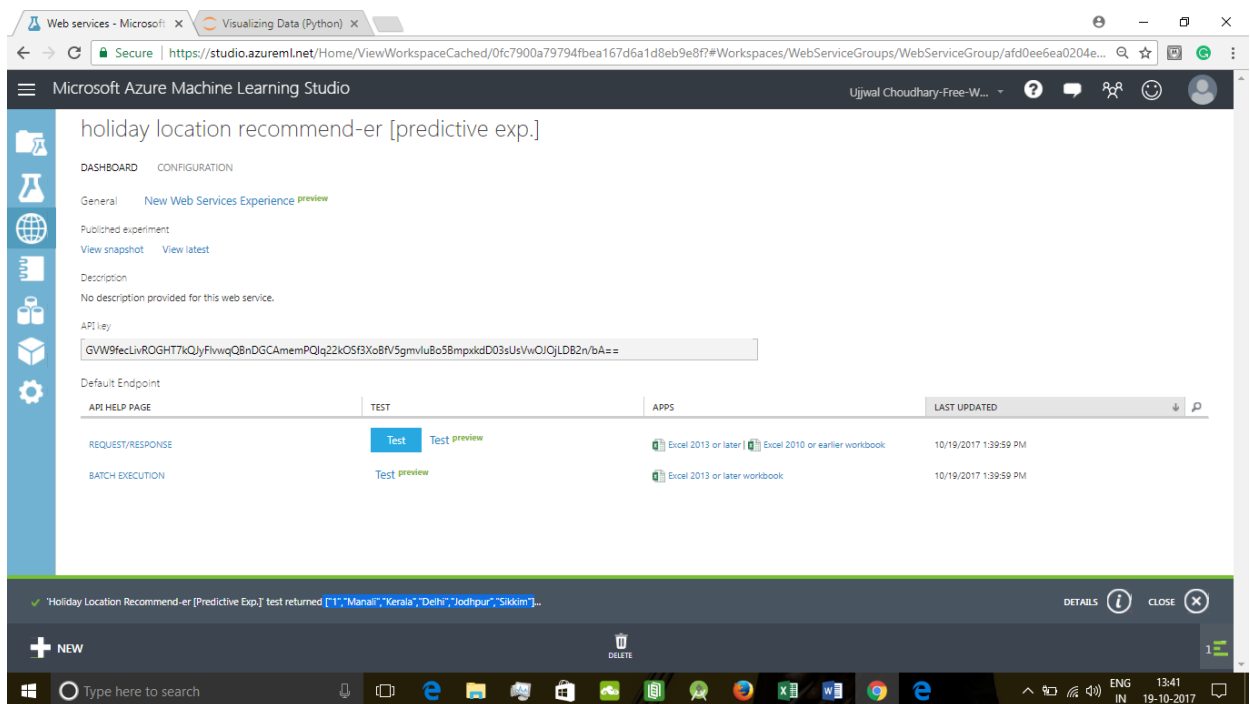
And Similarly for User ID 2 :-

For User Id :3