JOB-A-THON-JUNE-2021

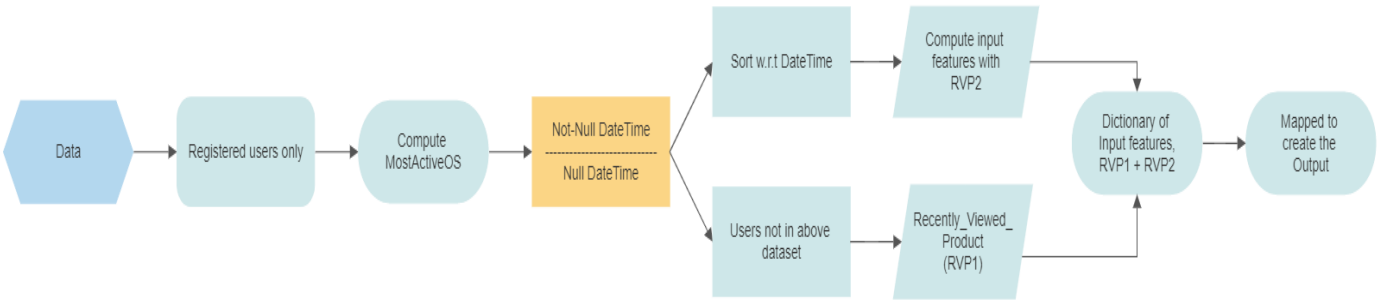
Introduction

This dataset aims to solve the problem of finding the customer who would like to take interest in taking the product. The data contains the details of customers who all are eligible for purchasing products.

Aim:

We are expected to develop input features from the dataset using data engineering techniques. The purpose to build a set of features that could further be used to feed the ML models to generate predictions to fuel the personalized advertisements, email marketing campaigns, or special offers on the landing and category pages of the company's website.

Architecture:



Approach:

My approach for the problem was simple. First, I filtered out the relevant column features I need to generate the input features. After that, I checked for the null values. The four most important features

which had the null values and need to be imputed properly were the **VisitDateTime, UserID, ProductID and Activity.** Once we are done with imputing the null values, it's smooth sailing from thereon. We just need to compute a dictionary with all the input feature values with **UserID** as key and map it to the sample_submission file. That got me to be in Top-100 with a score of 0. 913167798907148

UserID: Here we are trying to impute null values using **webClientID**. The idea is users would have the same **webClientID** unless they clear their browser cookie. Well it turned out not to be helpful. We find out that all the null values are associated with the unregistered users. So removing all rows with the null here was the right way to go.

VisitDateTime: There wasn't any way to impute the date and time for this null feature. However, it didn't pose any problem while computing several input features we needed like **User_Vintage** and **Most_Active_OS.**

So, it's better to filter out the null rows here too**.**

However, there were users with a very few records that would be totally filtered out totally. We need to be careful there as those would be needed to compute **Recently_Viewed_Product.**

ProductID: Creating two columns based on taking the difference of DateTime value. Taking difference from previous rows (back_diff) and the next rows (forward_diff).

Now we could use panda's forward and backward fill functions for null ProductID values in two separate columns (ffill_ProductID and bfill_ProductID).

The idea is where the ffill_ProductID is equal to bfill_ProductID, then impute any as they are same. When they aren't equal, we check the back_diff and forward_diff and impute whether ffill_ProductID or bfill_ProductID. The idea is if a ProductID is missing, it will likely be the preceding one if there is a very small difference in time with the preceding time compared to the following difference. Example- If I viewed product A and 5 mins later viewed another product with missing ID, then 1hr later viewed product B, then that missing value is likely to be a product A.

Activity: We just used panda's forward fill function followed by backward fill functions for null Activity values

Tools:

1. Pandas
2. Numpy
3. DateTime module

References:

Got a couple of ideas from reading the discussion section regarding the ffill() and bfill() function. The idea of taking time difference was mine which indeed helped me improve my score.

Score:- 0.913167798907148