Marketplace Feature Table --Approach

Tools Used - Pyspark

This is the approach I used to solved the problem.

Step-1) Load the user data and convert signup Date into Date format "yyyy-mm-dd"

Step-2) Load the visitor log data and preprocess to convert the date into date format and convert all strings to lower case.

Step-3) Dropping columns

a) Dropped web client ID, city and country as they add no useful information.

b) Dropped all rows where there is no UserID. Since this information is may be of just visitors who have not registered and we don't need this data.

Step-4) Imputing null values of visitor log data.

I noticed that User ID , product ID and Activity column has null values.

a) User ID - Fill the null values of visit date column with most visited date for each user. The logic behind filling the null values of this column is that the null values may belong to date that the user was most active on.

b) Product ID - Fill the product ID column based on UserID and visit date. I find the most viewed product for each visit date and for each user and I fill the null values with that product based on UserID and visit date. The logic is on any visit date the user might be viewing the same product which is viewed the most on that day.

There are still null values in the product ID column because there may be some users who does not have any product ID viewed information for any visit date. For those I fill the null values with top viewed product for each user.

c) Activity - Fill the activity column based on userID and product ID. I find the top most activity (Mode) for each product id for each user. So I fill the null values with that activity if there is product ID already present for that row for each user. The logic is for that product ID the user might be doing the same activity and hence I selected the activity which was done the most for that product ID.

If there are still null values where there is no product ID present for that row that I fill those cells with top most activity (Mode) for that user.

step-5) There will be still some null values left because there may be Users who have no Visit date , product ID and activity present in the cells. So these users might have not visited the site or there is no sufficient information for these users to fill the null values.

a) I fill the null values of the visit date column with top visited date for the entire table.

b) I fill the null values of product ID column with top viewed product for the entire table.

c) I fill the null values of activity column with top most activity for entire table.

Step-6) Create a column with datatype as timestamp from original VisitDateTime column which is in yyyy-mm-dd hh:mm:ss:SSS format. I have done it using CAST ("" as timestamp) to convert into timestamp format so that this column is used while creating the input feature, since we need to find the most recent date.

Step-7) I save the table as csv file with all preprocessing, so that it is easy to query from this csv file instead of loading all the preprocessing cells again if needed.

Step-8) Create Input feature table using Pyspark Dataframe

 # User Vintage - calculated the user vintage using Pyspark.sql.functions .datediff()

 # No. of days user visited in last 7 days –
Step-1) Filter the data where date is greater than 20-05-2018
Step-2) calculate the no.of days visited for each user using Spark SQL statement.
Step-3) Fill the null values with 0.

 # No. of products viewed in last 15 days –
Step-1) Filter the data where date is greater than 12-05-2018
Step-2) calculate the no.of products viewed for each user using Spark SQL statement.
Step-3) Fill the null values with 0.

 # Most viewed product in the last 15 days –
Step-1) Filter the data where date is greater than 12-05-2018 and where activity == pageload
Step-2) extracted the most recent timestamp for each user and for each product id.
Step-3) Join the two tables and calculated the product id which is most visited and if user has multiple products with same views then extracted product id which is most recent as per the timestamp.
Step-4) Fill the null values with "Product101".
Note: I observed that this timestamp is important because there are multiple users who have same no. of views for multiple products and also have same date. So hours and minutes information is crucial to extract the correct data.

 # Most frequent used OS – Using Spark SQL statement extracted the most frequent used OS by the user

 # Most recent viewed product by the user –
Step-1) Filter the data where activity == pageload and group by userID and product id and timestamp.
Step-2) sort the table in desc order based on timestamp.
Step-3) Using aggregate function in Pyspark extract the most recent viewed product for each user.
Step-4) Fill the null values with "Product101".

 # No. of pageload in last 7 days –
Step-1) filter the data where date is greater than 20-05-2018 and where activity == pageload.
Step-2) Count the no. of page loads using Spark SQL statement.
Step-3) Fill the null values with 0.

 # No. of clicks in last 7 days –
Step-1) filter the data where date is greater than 20-05-2018 and where activity == click.

Step-2) Count the no. of page loads using Spark SQL statement.
Step-3) Fill the null values with 0.

.