Table of Contents

[Revision History 2](#_Toc530573567)

[Introduction 3](#_Toc530573568)

[Connect to development environment (R/python/spark ) 3](#_Toc530573569)

[ Download google cloud sdk (command line tools) 3](#_Toc530573570)

[ You will have to authenticate to the google account login 3](#_Toc530573571)

[ Once you have authenticated you should be able to ssh tunnel 3](#_Toc530573572)

[ Now you should be able to launch jupyter 3](#_Toc530573573)

[ If the image is not up, you can run the following command 3](#_Toc530573574)

[ Now go to tab directory work/srv/dev. 4](#_Toc530573575)

[Feature Extraction : 4](#_Toc530573576)

[ Connect to bigQuery 4](#_Toc530573577)

[ High level Data analysis 5](#_Toc530573578)

[ Sales analysis. 6](#_Toc530573579)

[I. Daily sales by Store. 6](#_Toc530573580)

[II. Weekly subtotal 7](#_Toc530573581)

[III. Monthly seasonality 8](#_Toc530573582)

[IV. State level sales 9](#_Toc530573583)

[V. Store/month and state level seasonality. 10](#_Toc530573584)

[ Finding the anomalies in data 11](#_Toc530573585)

[ Analysis at VIN level 11](#_Toc530573586)

[ Analysis at Bays level 11](#_Toc530573587)

[ Analysis at store Level 12](#_Toc530573588)

[ Product level Analysis 12](#_Toc530573589)

[ Product/sales level analysis 13](#_Toc530573590)

[I. Store and Date wise avg subtotal/discount/profit 13](#_Toc530573591)

[ Cluster characteristics 14](#_Toc530573592)

[ Weather data feed-up 15](#_Toc530573593)

[ Competitor level analysis 15](#_Toc530573594)

[I. Direct Competitor with in 5 miles 15](#_Toc530573595)

[II. Indirect Competitor with in 5 miles 15](#_Toc530573596)

[ Vehicles data 16](#_Toc530573597)

[I. Vehicles /Car count within 5 miles of our stores 16](#_Toc530573598)

[ Market stores saturation 17](#_Toc530573599)

[ Competitor pricing Information 17](#_Toc530573600)

[ Holiday Data 18](#_Toc530573601)

[ Final View 18](#_Toc530573602)

[Importing required data/Data Frame. 21](#_Toc530573603)

[ Import data in import data module tab 21](#_Toc530573604)

[Add\_regressor/feature to fit the model 22](#_Toc530573605)

[ Current regressor. 22](#_Toc530573606)

[ Adding new regressor/feature 22](#_Toc530573607)

[Validating regressor in model training. 25](#_Toc530573608)

[Important Query 26](#_Toc530573609)

[Import Tables (Do not Delete ) 27](#_Toc530573610)

[How to Restart google server 27](#_Toc530573611)

# Revision History

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Date | Created By | Review By | Version | Remark |
| 30-Oct-2018 | Kaptan Singh |  | Draft 0.1 | Initiated Draft version |
| 03-Nov-2018 | Kaptan Singh |  | Draft 1.0 | Updated with feature extraction |

# Introduction

The purpose of document is to explain high level of data flow in the shiny app. Feature extraction ,adding the feature to model modify the data or how we can load data to Shinny platform , add features /extra regressor in model training function , add regressor to all the related/impacted tabs like global.r , import data , forecasting, optimization tabs then validating the regressor/model with newly added features . Re-stating and taking backup of server.

# Connect to development environment (R/python/spark )

## Download google cloud sdk (command line tools)

* 1. <https://cloud.google.com/sdk/docs/>
  2. ./google-cloud-sdk/install.sh
  3. ./google-cloud-sdk/bin/gcloud init

## You will have to authenticate to the google account login

## Once you have authenticated you should be able to ssh tunnel

* 1. Save ssh file in directory preferably in .ssh (ssh file is in teams files: google\_compute\_engine"). For Windows putty users, use google.ppk.
  2. Local terminal run
     1. ssh -i .ssh/google\_compute\_engine.pub -L 8888:localhost:8888 -f -N [jm186069@](mailto:jm186069@104.196.221.171)35.196.174.53 (use your own id )

## Now you should be able to launch jupyter

* 1. localhost:8888/?token=68b12ca49d158fee935f484802400e7a315f727752f0b01b (token use to change whenever refresh docker)
  2. If you prefer lab then you can go to localhost:8888/lab
  3. If jupyter is not running you can go to console and see if docker images is up

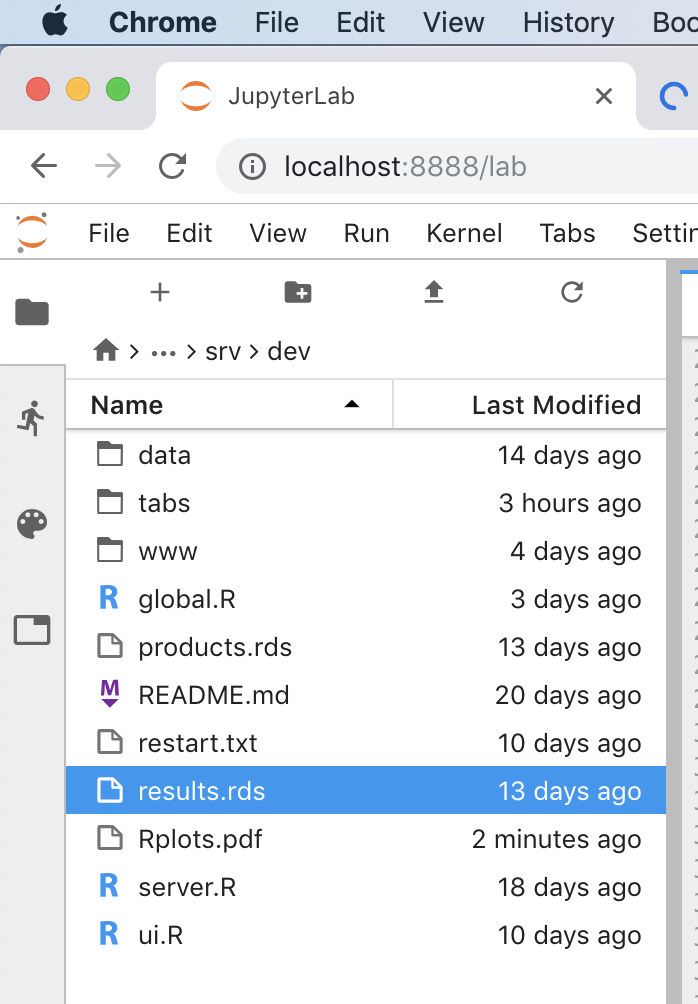
Docker ps

## If the image is not up, you can run the following command

* 1. docker run --user root -e GRANT\_SUDO=yes -it --rm -p 8888:8888 -v /home/jack\_mccush/test:/home/jovyan/work jupyter/all-spark-notebook

## Now go to tab directory work/srv/dev.

once you able to run above command successfully you will able to see below window



# Feature Extraction :

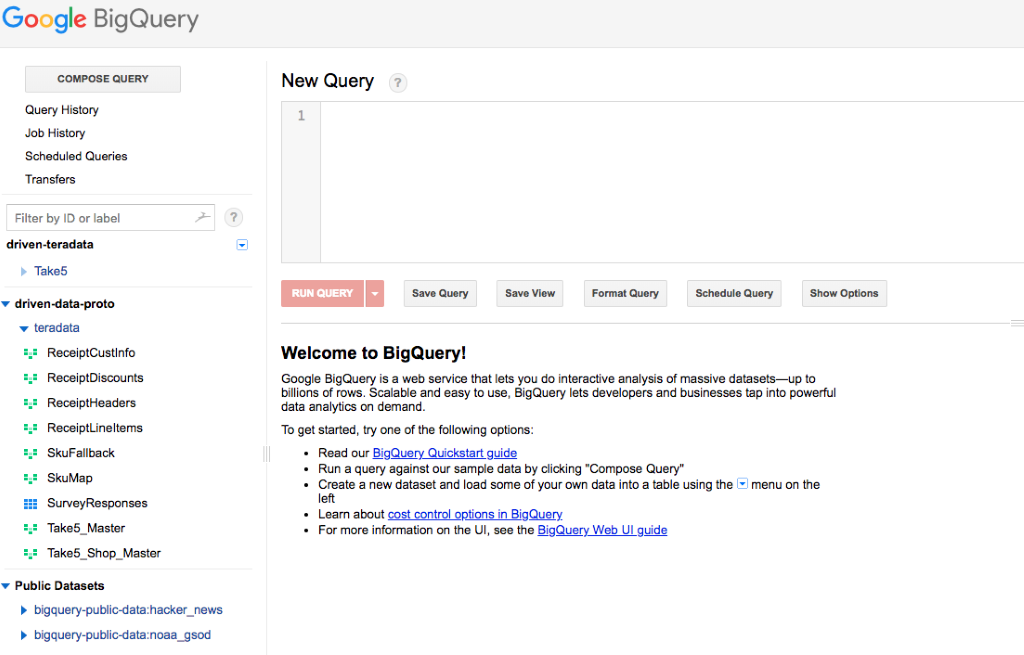
Feature extraction involves reducing the amount of resources required to describe a large set of data. When performing analysis of complex data one of the major problems stems from the number of variables involved. Analysis with a large number of variables generally requires a large amount of memory and computation power and it may cause model overfitting

feature extraction is very important it starts from an initial set of measured data and builds derived values ([features](https://en.wikipedia.org/wiki/Feature_(machine_learning))) Feature extraction is a [dimensionality reduction](https://en.wikipedia.org/wiki/Dimensionality_reduction) process, where an initial set of raw variables is reduced to more manageable groups (features) for processing,

When the input data to an [algorithm](https://en.wikipedia.org/wiki/Algorithm) is too large to be processed and it is suspected to be redundant then it can be transformed into a [feature vector](https://en.wikipedia.org/wiki/Feature_vector). Determining a subset of the initial features is called features extraction.

## Connect to bigQuery

1. Create a net Google Account with your Teradata email
   1. (be sure not to create @gmail address (use Teradata email as account)
2. Join Driven Brands google groups
   1. <https://groups.google.com/forum/#!forum/driven-teradata>
3. Once you have been granted access, you will need to active gcloud account
   1. I had to input credit card to activate account.
4. Now you should have access to the big query table  go to following link to write query
   1. <https://bigquery.cloud.google.com/welcome/driven-teradata?utm_source=bqux&utm_medium=link&utm_campaign=beta>



## High level Data analysis

High level analysis includes information’s at store level, data duration , products etc , here we are using forecasting model so we need add time serious data. We are going to analgise data after date Jan – 2018.

Note data point may be vary due to daily data updates in tables

#SELECT count(\*) FROM `driven-data-proto.teradata.ReceiptHeaders` ----10485413

#SELECT count(DISTINCT(StoreNum)) FROM `driven-data-proto.teradata.ReceiptHeaders` ----314

#SELECT count(distinct(ZipCode)) FROM `driven-data-proto.teradata.ReceiptHeaders` ----31703

#SELECT sum(SubTotal) as sub\_Total\_sum , ZipCode  FROM `driven-data-proto.teradata.ReceiptHeaders` where sub\_Total\_sum <> 0 group by ZipCode

#SELECT count(distinct(ZIP)) FROM `driven-data-proto.teradata.Take5\_Master` --27329

#SELECT count(\*) FROM `driven-data-proto.teradata.Take5\_Master` group by ZIP

#SELECT distinct(State) FROM `driven-data-proto.teradata.Take5\_Master`----65

#select count(\*) as CITY\_UNDER\_STATE, CITY, State , ZIP   from `driven-data-proto.teradata.Take5\_Master` where state in ( SELECT distinct(State) FROM `driven-data-proto.teradata.Take5\_Master`) group by CITY, State, ZIP

select count(\*) as CITY\_UNDER\_STATE, CITY, State   from `driven-data-proto.teradata.Take5\_Master`   
where state in ( SELECT distinct(State) FROM `driven-data-proto.teradata.Take5\_Master`)  
#and CITY\_UNDER\_STATE [NOT] IN ('1')  
group by CITY, State

* To get the total amount  based on  date. (assuming we have some -/+ amt).

SELECT sum(RH.SubTotal) as SubTotal,sum(RH.Total)  as Total ,  RH.ReceiptDate FROM `driven-data-proto.teradata.ReceiptHeaders` as RH where RH.ReceiptDate between '2018-08-01' AND '2018-08-05' group by RH.ReceiptDate order by RH.ReceiptDate

* To get discountpercent and discountAMT based on date and totalAmt.

SELECT sum(RH.SubTotal) as SubTotal,sum(RH.Total)  as Total ,  RH.ReceiptDate , RD.DiscountPercent, RD.DiscountAMT , RH.ReceiptNum, TM.CITY  
FROM `driven-data-proto.teradata.ReceiptDiscounts` AS RD  
JOIN `driven-data-proto.teradata.ReceiptHeaders` as RH ON RD.ReceiptID=RH.ReceiptID  
JOIN `driven-data-proto.teradata.Take5\_Master`  as TM ON RH.ReceiptID=TM.ReceiptID  
where RH.ReceiptDate between '2018-08-01' AND '2018-08-05' group by RH.ReceiptDate, RD.DiscountPercent, RH.ReceiptNum,RD.DiscountAMT, TM.CITY  
order by RH.ReceiptDate

* To get the maximum Total/ discountAmt based on city and storeNum.

 SELECT max(Total) as Total , CITY,StoreNum   FROM ( SELECT sum(RH.SubTotal) as SubTotal,sum(RH.Total)  as Total ,  RH.ReceiptDate , RD.DiscountPercent, RD.DiscountAMT , RH.ReceiptNum, TM.CITY,RH.StoreNumFROM `driven-data-proto.teradata.ReceiptDiscounts` AS RD  
JOIN `driven-data-proto.teradata.ReceiptHeaders` as RH ON RD.ReceiptID=RH.ReceiptID  
JOIN `driven-data-proto.teradata.Take5\_Master`  as TM ON RH.ReceiptID=TM.ReceiptID  
where RH.ReceiptDate between '2018-08-01' AND '2018-08-05' group by RH.ReceiptDate, RD.DiscountPercent, RH.ReceiptNum,RD.DiscountAMT, TM.CITY,RH.StoreNum  
order by RH.ReceiptDate)  
group by CITY,StoreNum

SELECT max(DiscountAMT) as DiscountAMT , CITY,StoreNum   FROM ( SELECT sum(RH.SubTotal) as SubTotal,sum(RH.Total)  as Total ,  RH.ReceiptDate , RD.DiscountPercent, RD.DiscountAMT , RH.ReceiptNum, TM.CITY,RH.StoreNum  
FROM `driven-data-proto.teradata.ReceiptDiscounts` AS RD  
JOIN `driven-data-proto.teradata.ReceiptHeaders` as RH ON RD.ReceiptID=RH.ReceiptID  
JOIN `driven-data-proto.teradata.Take5\_Master`  as TM ON RH.ReceiptID=TM.ReceiptID  
where RH.ReceiptDate between '2018-08-01' AND '2018-08-05' group by RH.ReceiptDate, RD.DiscountPercent, RH.ReceiptNum,RD.DiscountAMT, TM.CITY,RH.StoreNum  
order by RH.ReceiptDate)  
group by CITY,StoreNum

## Sales analysis.

We did some analysis sales data at daily level /weekly/monthly/yearly and state wise . The idea was to understand do we have any seasonality in sale or not.

### Daily sales by Store.

SELECT CAST(avg(RH.SubTotal) AS NUMERIC) as Avgprice,

CAST(sum(RH.SubTotal) AS NUMERIC) as TotalSale,

CAST(sum(RD.DiscountAMT ) AS NUMERIC) as TotalDiscountAMT,

CAST(sum(TM.GrossProfit) as NUMERIC) as TotalGrossProfit,

TM.storeNum,

CAST(count(TM.ReceiptID ) AS NUMERIC) as TotalReceiptID,

#EXTRACT(month FROM TM.ReceiptDate) As Month,

#EXTRACT(Year FROM TM.ReceiptDate) As Year,

TM.ReceiptDate

from `driven-data-proto.teradata.Take5\_Master` AS TM

JOIN `driven-data-proto.teradata.ReceiptHeaders` AS RH

on TM.ReceiptID= RH.ReceiptID

JOIN `driven-data-proto.teradata.ReceiptDiscounts` as RD

ON RD.ReceiptID=TM.ReceiptID

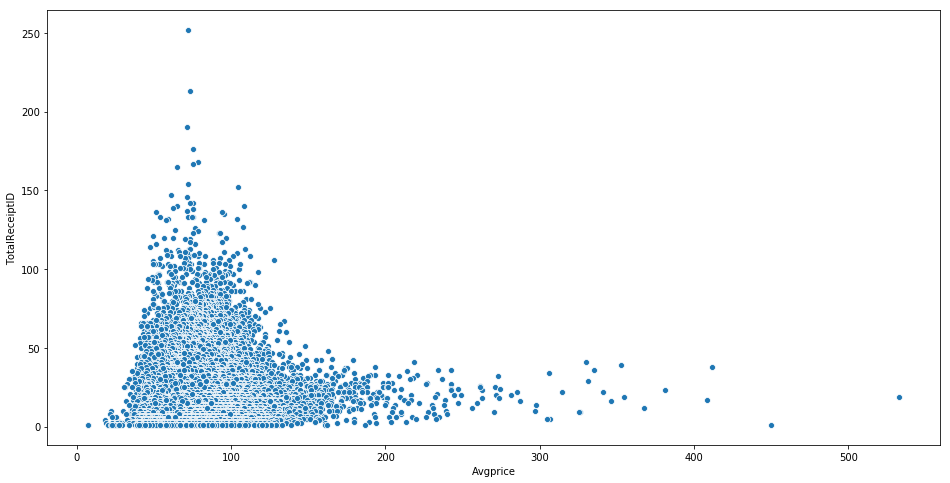
where TM.ReceiptDate between '2018-01-01' AND '2018-08-31'

and TM.storeNum Not in (SELECT StoreNum FROM `driven-teradata.Andrew.BadStoreNum` )

and TM.VIN Not in (SELECT vin FROM `driven-teradata.Andrew.BadVINs` )

group by TM.storeNum, TM.ReceiptDate

order by TotalGrossProfit desc



### Weekly subtotal

We are taking date between '2016-09-01' AND '2017-08-31' and '2017-10-01' and '2018-9-30' to check the sales difference in years.

Note change date as per requirement.

SELECT  CAST(avg(RH.SubTotal) AS INT64) as AvgSale

, CAST(avg(RD.DiscountAMT ) AS INT64) as TotalDiscountAMT

, CAST(avg(TM.GrossProfit) as INT64) as TotalGrossProfit

#, EXTRACT(month FROM TM.ReceiptDate) As Month

, EXTRACT(WEEK FROM TM.ReceiptDate) AS week

, count(rh.ReceiptID) as CuntOFReceiptID

from `driven-data-proto.teradata.Take5\_Master`  AS TM

JOIN `driven-data-proto.teradata.ReceiptHeaders`  AS RH

on (TM.ReceiptID= RH.ReceiptID

and CAST(TM.ReceiptDate AS STRING) = RH.ReceiptDate

)

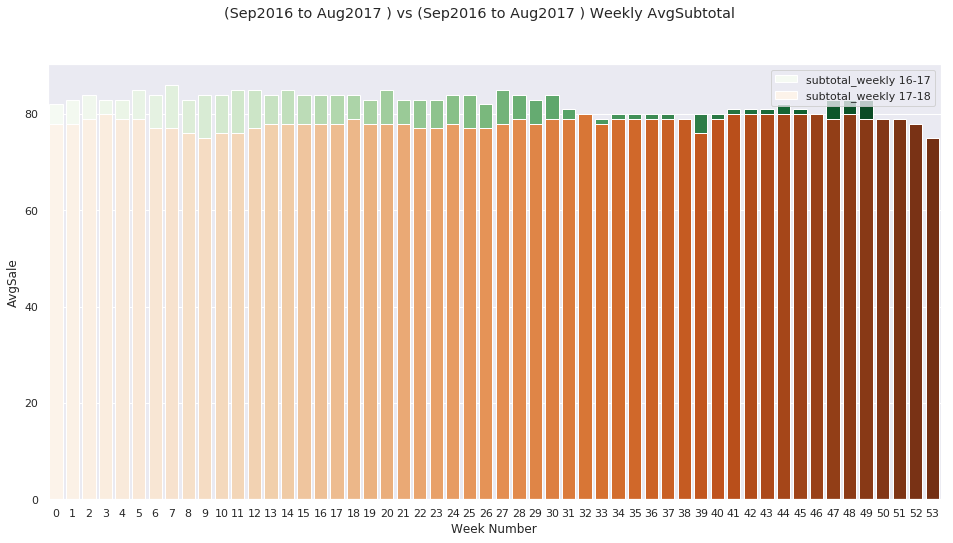
JOIN `driven-data-proto.teradata.ReceiptDiscounts` as  RD

ON RD.ReceiptID=TM.ReceiptID

where TM.ReceiptDate between '2017-10-01' AND '2018-9-30'

group by week

order by week



### Monthly seasonality

SELECT  CAST(avg(RH.SubTotal) AS INT64) as AvgSubtotal

, CAST(avg(RD.DiscountAMT ) AS INT64) as TotalDiscountAMT

, CAST(avg(TM.GrossProfit) as INT64) as TotalGrossProfit

, EXTRACT(month FROM TM.ReceiptDate) As Month

, CASE EXTRACT(month FROM TM.ReceiptDate)

WHEN 1 THEN 'JAN18'

WHEN 2 THEN 'FEB18'

WHEN 3 THEN 'MAR18'

WHEN 4 THEN 'APR18'

WHEN 5 THEN 'MAY18'

WHEN 6 THEN 'JUN18'

WHEN 7 THEN 'JUL18'

WHEN 8 THEN 'AUG17'

WHEN 9 THEN 'SEP17'

WHEN 10 THEN 'OCT17'

WHEN 11 THEN 'NOV17'

WHEN 12 THEN 'DEC17'

END

#, EXTRACT(month FROM TM.ReceiptDate) As Month

#, EXTRACT(WEEK FROM TM.ReceiptDate) AS week

, count(rh.ReceiptID) as CuntOFReceiptID

from `driven-data-proto.teradata.Take5\_Master`  AS TM

JOIN `driven-data-proto.teradata.ReceiptHeaders`  AS RH

on (TM.ReceiptID= RH.ReceiptID

and CAST(TM.ReceiptDate AS STRING) = RH.ReceiptDate

)

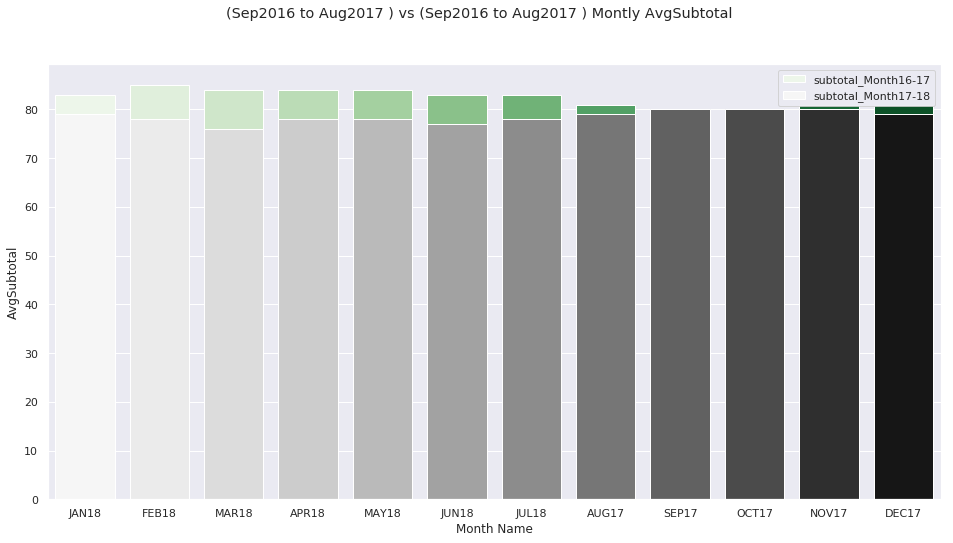
JOIN `driven-data-proto.teradata.ReceiptDiscounts` as  RD

ON RD.ReceiptID=TM.ReceiptID

where TM.ReceiptDate between '2017-09-01' AND '2018-08-31'

group by 4,5

order by 4



### State level sales

SELECT  CAST(avg(RH.SubTotal) AS INT64) as AvgSubtotal

, CAST(avg(RD.DiscountAMT ) AS INT64) as TotalDiscountAMT

, CAST(avg(TM.GrossProfit) as INT64) as TotalGrossProfit

,(Tm.State)

#, EXTRACT(month FROM TM.ReceiptDate) As Month

#, EXTRACT(WEEK FROM TM.ReceiptDate) AS week

, count(rh.ReceiptID) as CuntOFReceiptID

from `driven-data-proto.teradata.Take5\_Master`  AS TM

JOIN `driven-data-proto.teradata.ReceiptHeaders`  AS RH

on (TM.ReceiptID= RH.ReceiptID

and CAST(TM.ReceiptDate AS STRING) = RH.ReceiptDate

)

JOIN `driven-data-proto.teradata.ReceiptDiscounts` as  RD

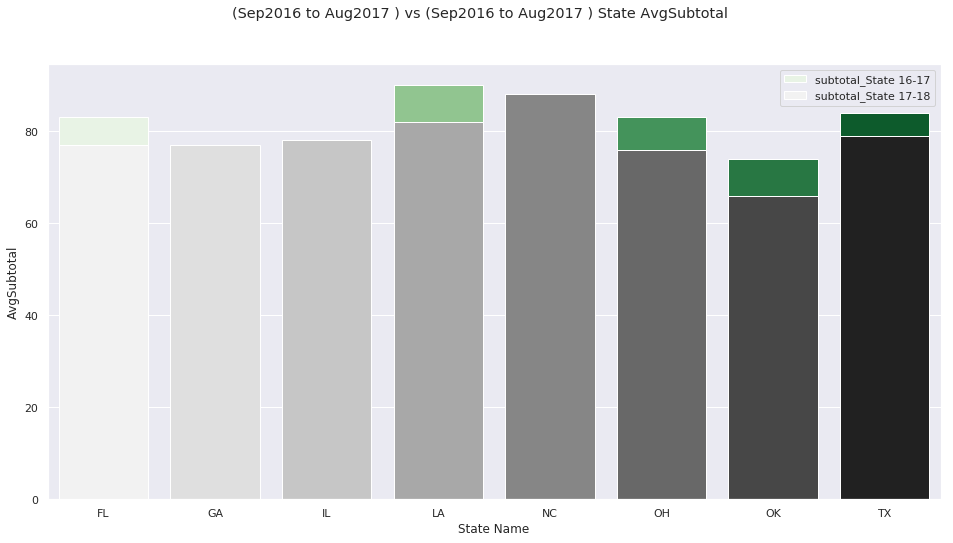
ON RD.ReceiptID=TM.ReceiptID

where TM.ReceiptDate between '2017-09-01' AND '2018-08-31'

and Tm.State in ('GA', 'OK', 'NC', 'IL', 'LA', 'OH', 'FL', 'TX')

group by Tm.State

order by Tm.State



### Store/month and state level seasonality.

Please modify query for all other states.

 SELECT tm.state,

CAST(avg(RH.SubTotal) AS INT64) as AvgSubtotal

, CAST(avg(RD.DiscountAMT ) AS INT64) as AvgDiscountAMT

#, EXTRACT(month FROM TM.ReceiptDate) As Month

, substr(rh.receiptdate, 1, 7) as month

, count(rh.ReceiptID) as CountOFReceiptID

, round(avg(RH.SubTotal) \* count(rh.receiptid), 0) as TotalStateSales

, round((avg(RH.SubTotal) \* count(rh.receiptid)) / count(distinct rh.storenum), 0) as AvgStoreSales

, count(distinct rh.storenum) as CountOfStores

from `driven-data-proto.teradata.Take5\_Shop\_Master` AS TM,

`driven-data-proto.teradata.ReceiptHeaders` rh,

(

select storenum from `driven-data-proto.teradata.ReceiptHeaders` rh

group by 1

having min(ReceiptDate) <= '2017-09-01' and max(ReceiptDate) >= '2018-08-31'

) AS QualStores

left join `driven-data-proto.teradata.ReceiptDiscounts` as RD

on RD.ReceiptID=RH.ReceiptID

where rh.storenum = qualstores.storenum

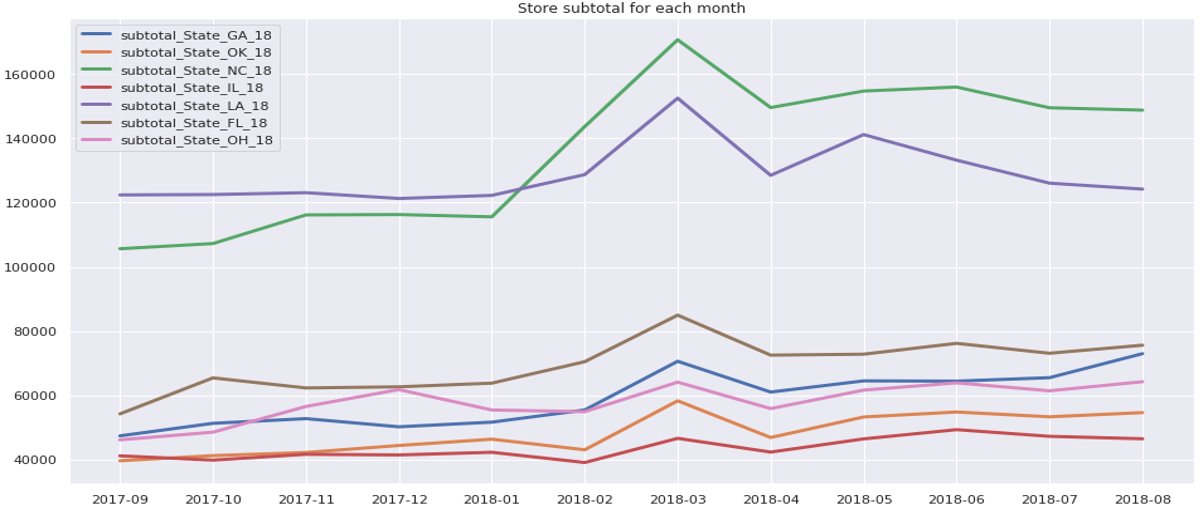
and (tm.storenum = rh.storenum)

and rh.ReceiptDate between '2016-09-01' AND '2017-08-31'

and Tm.State in ('FL')

#('OK', 'NC', 'IL', 'LA', 'OH', 'FL', 'TX')

group by 1,4 order by 1,4



## Finding the anomalies in data

Now at this level we are planning to find out the anomalies in the data at store level to find out the anomalies please run below query

SELECT storenum, MIN(CAST(receiptdate AS date)),  MAX(CAST(receiptdate AS date)), DATE\_DIFF(MAX(CAST(receiptdate AS date)), MIN(CAST(receiptdate AS date)), DAY), COUNT(ReceiptID) FROM   `driven-data-proto.teradata.ReceiptHeaders` GROUP BY   1 ORDER BY  5 DESC

we're mostly looking at 8/2018, most of these will be excluded automatically, but we should be sure to exclude storenum 240 from analysis.

Created table with storenums to exclude from analysis:

create table`driven-teradata.Andrew.BadStoreNum` as SELECT storenum  
FROM   `driven-data-proto.teradata.ReceiptHeaders` GROUP BY  1  
having  DATE\_DIFF(MAX(CAST(receiptdate AS date)), MIN(CAST(receiptdate AS date)), DAY) < 50 or max(cast(receiptdate as date)) < '2018-01-01'

## Analysis at VIN level

Analysis by customer should exclude null VIN, all-space VIN, and all 1s, 2s, 9s, and 0s , advise to create a another table.

SELECT  vin, MIN(CAST(receiptdate AS date)), MAX(CAST(receiptdate AS date)),  
  DATE\_DIFF(MAX(CAST(receiptdate AS date)), MIN(CAST(receiptdate AS date)), DAY), COUNT(ReceiptID) FROM   `driven-data-proto.teradata.ReceiptHeaders`  
GROUP BY   1 ORDER BY  5 DESC limit 100

## Analysis at Bays level

Here we can check Average sales by number of bays.

SELECT NumberOfBays,  AVG(AvgDailySales) FROM (  SELECT   SM.NumberOfBays, SM.StoreNum,  
   SUM(RH.Total) / DATE\_DIFF(MAX(CAST(receiptdate AS date)), MIN(CAST(receiptdate AS date)), DAY) AS AvgDailySales FROM    `driven-data-proto.teradata.ReceiptHeaders` AS RH,  `driven-data-proto.teradata.Take5\_Shop\_Master` AS SM  LEFT JOIN    `driven-teradata.Andrew.BadStoreNum` AS BadShop  ON    RH.StoreNum = BadShop.storenum   WHERE     BadShop.StoreNum IS NULL     AND RH.StoreNum = SM.StoreNum     AND RH.ReceiptDate BETWEEN '2018-08-01'    AND '2018-08-05'  
  GROUP BY   1,    2) a GROUP BY   1 ORDER BY 1

## Analysis at store Level

select  
  avgsalesbycatbyday.storenum,  
  max(case when productsubgroup = 'HIGHMILE' then avgtranscount else 0 end) as highmile,  
  max(case when productsubgroup = 'SYNBLEND' then avgtranscount else 0 end) as synblend,  
  max(case when productsubgroup = 'SYNTHETIC' then avgtranscount else 0 end) as synthetic,  
  max(case when productsubgroup = 'CONVENTIONAL' then avgtranscount else 0 end) as conventional,  
  max(case when productsubgroup = 'AIR FILTER' then avgtranscount else 0 end) as airfilter,  
  max(case when productsubgroup = 'CABIN FILTER' then avgtranscount else 0 end) as cabinfilter,  
  max(case when productsubgroup = 'WIPER BLADES' then avgtranscount else 0 end) as wiperblades  
from  
(select storenum, productsubgroup, avg(transcount) as avgtranscount from  (SELECT storenum, productsubgroup, receiptdate, count(\*) as transcount  
  FROM `driven-data-proto.teradata.ReceiptLineItems` rl,  
  `driven-data-proto.teradata.ReceiptHeaders` rh,  `driven-data-proto.teradata.SkuMap` sku  
  where sku.productsku = concat(rtrim(rl.productcode), '-', rl.service)   and rl.receiptid rh.receiptid  and productsubgroup <> 'MISC'  group by 1,2,3) salescatbyday  group by 1,2) avgsalesbycatbyday left join `driven-teradata.Andrew.BadStoreNum` bsn  
on bsn.storenum = avgsalesbycatbyday.storenum where bsn.storenum is null group by 1;

## Product level Analysis

Here we started at product level analysis, the idea behind to understand products Group and products sub group and product code.

Data at product Group level.

select count(ProductGroup) as ProductGroupA, ProductGroup from (

SELECT CAST(avg(RL.LineItemID ) AS NUMERIC) as SumofLineItem,

CAST(avg(RH.SubTotal) AS NUMERIC) as Avgprice,

CAST(sum(RH.SubTotal) AS NUMERIC) as TotalSale,

CAST(sum(RD.DiscountAMT ) AS NUMERIC) as TotalDiscountAMT,

CAST(sum(TM.GrossProfit) as NUMERIC) as TotalGrossProfit,

TM.storeNum,

CAST(count(TM.ReceiptID ) AS NUMERIC) as TotalReceiptID,

#RL.ProductCode,

count(SF.ProductGroup) as countProductGroup ,

SF.ProductGroup,

#EXTRACT(month FROM TM.ReceiptDate) As Month,

#EXTRACT(Year FROM TM.ReceiptDate) As Year,

TM.ReceiptDate

from `driven-data-proto.teradata.Take5\_Master` AS TM

JOIN `driven-data-proto.teradata.ReceiptHeaders` AS RH

on TM.ReceiptID= RH.ReceiptID

JOIN `driven-data-proto.teradata.ReceiptDiscounts` as RD

ON RD.ReceiptID=TM.ReceiptID

JOIN `driven-data-proto.teradata.ReceiptLineItems` as RL

ON RL.ReceiptID=TM.ReceiptID

JOIN `driven-data-proto.teradata.SkuFallback` as SF

on CAST(RL.ProductCode AS NUMERIC) = SF.ProductCode

where TM.ReceiptDate between '2018-01-01' AND '2018-08-31'

and TM.storeNum Not in (SELECT StoreNum FROM `driven-teradata.Andrew.BadStoreNum` )

and TM.VIN Not in (SELECT vin FROM `driven-teradata.Andrew.BadVINs` )

group by TM.storeNum, TM.ReceiptDate , SF.ProductGroup

order by TotalGrossProfit desc ) group by ProductGroup order by ProductGroupA desc

data at product sub group level

select count(ProductSubGroup) as ProductSubGroupA , ProductSubGroup,ProductGroup , ProductCode from (  
SELECT  CAST(avg(RL.LineItemID ) AS NUMERIC) as SumofLineItem,  
CAST(avg(RH.SubTotal) AS NUMERIC) as Avgprice,  
CAST(sum(RH.SubTotal) AS NUMERIC) as TotalSale,  
CAST(sum(RD.DiscountAMT ) AS NUMERIC) as TotalDiscountAMT,  
CAST(sum(TM.GrossProfit) as NUMERIC) as TotalGrossProfit,  
TM.storeNum,  
CAST(count(TM.ReceiptID ) AS NUMERIC) as TotalReceiptID,  
RL.ProductCode,  
count(SF.ProductGroup) as countProductGroup ,  
SF.ProductGroup,  
#EXTRACT(month FROM TM.ReceiptDate) As Month,  
#EXTRACT(Year FROM TM.ReceiptDate) As Year,  
count(SF.ProductSubGroup ) as countProductSubGroup,  
SF.ProductSubGroup ,  
TM.ReceiptDate  
from `driven-data-proto.teradata.Take5\_Master`  AS TM  
JOIN `driven-data-proto.teradata.ReceiptHeaders`  AS RH  
on TM.ReceiptID= RH.ReceiptID  
JOIN `driven-data-proto.teradata.ReceiptDiscounts` as  RD  
ON RD.ReceiptID=TM.ReceiptID  
JOIN `driven-data-proto.teradata.ReceiptLineItems` as  RL  
ON RL.ReceiptID=TM.ReceiptID  
JOIN `driven-data-proto.teradata.SkuFallback` as  SF  
on CAST(RL.ProductCode AS NUMERIC) = SF.ProductCode  
where TM.ReceiptDate between '2018-01-01' AND '2018-08-31'  
and  TM.storeNum Not in (SELECT StoreNum FROM `driven-teradata.Andrew.BadStoreNum` )  
and TM.VIN Not in (SELECT vin FROM `driven-teradata.Andrew.BadVINs` )  
group by TM.storeNum, TM.ReceiptDate , RL.ProductCode, SF.ProductGroup, SF.ProductSubGroup order by TotalGrossProfit  desc )  
group by ProductSubGroup, ProductGroup, ProductCode  
order by ProductSubGroupA desc

## Product/sales level analysis

prod sub group level profit/margin

SELECT avg( li.LineItemTotal ) as AvgLineItemTotal,  
        avg(Li.GrossProfit) as AvgGrossProfit,  
        avg(li.GrossMargin) as AvgGrossMargin,  
        count(rh.ReceiptID) as count\_of\_ReceiptID,  
        sm.ProductSubGroup,  rh.StoreNum,    rh.ReceiptDate,  
        count(rh.ReceiptID)  as countOfReceiptID  
        FROM `driven-data-proto.teradata.ReceiptHeaders` rh  
          INNER JOIN (( select ReceiptID, ProductCode,Service , LineItemTotal,  
          round(LineItemTotal - TotalCOGS,2) as GrossProfit,  
          round(SAFE\_DIVIDE((LineItemTotal - TotalCOGS), LineItemTotal),2) as GrossMargin  
          from  
                (  
                select \* round(( PartsCOGS + LubricantCOGS),2) as TotalCOGS  
                from `driven-data-proto.teradata.ReceiptLineItems`                )  )  
                ) li   on rh.ReceiptID = li.ReceiptID  
          INNER JOIN `driven-data-proto.teradata.SkuMap` sm   On Concat(TRIM(li.ProductCode), '-', TRIM(li.Service)) = sm.ProductSKU  
where  
RH.ReceiptDate  between '2018-01-01' AND '2018-09-20'  
and sm.ProductSubGroup in ('SYNTHETIC' )  
      Group by sm.ProductSubGroup,   rh.StoreNum,  
              rh.ReceiptDate,    li.LineItemTotal

### Store and Date wise avg subtotal/discount/profit

SELECT avg(Rh.subtotal) as AvgSubtotal  
, avg(RD.DiscountAMT) as AvgDiscountAMT  
, avg(tm.GrossProfit ) as AvgTMGrossProfit  
, avg(Tm.GrossMargin ) as AvgTMGrossMargin  
, rh.storenum  
, count(rh.ReceiptID ) as countReceiptID  
, rh.ReceiptDate  
FROM `driven-data-proto.teradata.ReceiptHeaders`  Rh  
JOIN `driven-data-proto.teradata.Take5\_Master` TM  ON (rh.ReceiptID= TM.ReceiptID  AND CAST(RH.ReceiptDate  AS DATE) = TM.ReceiptDate AND rh.storenum = TM.storenum  )  
LEFT JOIN `driven-data-proto.teradata.ReceiptDiscounts`  RD ON RD.ReceiptID=RH.ReceiptID  
where RH.ReceiptDate  between '2018-01-01' AND '2018-09-20'  
#AND rh.storenum='0061'  
group by rh.storenum , rh.ReceiptDate

## Cluster characteristics

To do the clustering have identify the below characteristics.

select clustername,   
min(highmile) as minhighmile, max(highmile) as maxhighmile, avg(highmile) as avghighmile,  
min(synblend) as minsynblend, max(synblend) as maxsynblend, avg(synblend) as avgsynblend,  
min(synthetic) as minsynthetic, max(synthetic) as maxsynthetic, avg(synthetic) as avgsynthetic,  
min(conventional) as minconventional, max(conventional) as maxconventional, avg(conventional) as avgconventional,  
min(airfilter) as minairfilter, max(airfilter) as maxairfilter, avg(airfilter) as avgairfilter,  
min(cabinfilter) as mincabinfilter, max(cabinfilter) as maxcabinfilter, avg(cabinfilter) as avgcabinfilter,  
min(wiperblades) as minwiperblades, max(wiperblades) as maxwiperblades, avg(wiperblades) as avgwiperblades,  
min(totalt5sales) as mintotal, max(totalt5sales) as maxtotal, avg(totalt5sales) as avgtotal,  
count(\*) as numstores  
from  
(select storenum,   clustername,  
  max(case when productsubgroup = 'HIGHMILE' then avgtotal else 0 end) as highmile,  
  max(case when productsubgroup = 'SYNBLEND' then avgtotal else 0 end) as synblend,  
  max(case when productsubgroup = 'SYNTHETIC' then avgtotal else 0 end) as synthetic,  
  max(case when productsubgroup = 'CONVENTIONAL' then avgtotal else 0 end) as conventional,  
  max(case when productsubgroup = 'AIR FILTER' then avgtotal else 0 end) as airfilter,  
  max(case when productsubgroup = 'CABIN FILTER' then avgtotal else 0 end) as cabinfilter,  
  max(case when productsubgroup = 'WIPER BLADES' then avgtotal else 0 end) as wiperblades,   sum(avgtotal) as totalt5sales  
  from (select storenum, clustername, productsubgroup, avg(dailytotal) as avgtotal from  
(select s.storenum, clustername, numberofbays, receiptdate, productsubgroup, count(lineitemtotal) as dailytotal  
from `driven-teradata.t5analytics.ClusterNames` cn,  
`driven-teradata.t5analytics.StoreClusters` c,  
`driven-data-proto.teradata.Take5\_Shop\_Master` s,  
`driven-data-proto.teradata.ReceiptHeaders` rh,  
`driven-data-proto.teradata.ReceiptLineItems` rl,  
`driven-data-proto.teradata.SkuMap` sku  
WHERE s.storenum = c.storenum AND c.cluster = cn.cluster AND s.storenum = rh.storenum  
AND rh.receiptid = rl.receiptid AND sku.productsku = concat(rtrim(rl.productcode), '-', rl.service)  and sku.productsubgroup in ('HIGHMILE','SYNBLEND','SYNTHETIC','CONVENTIONAL','AIR FILTER','CABIN FILTER','WIPER BLADES') group by 1,2,3,4,5) avgsalesbyday group by 1,2,3) avgsalesbystore group by 1,2) group by 1

## Weather data feed-up

As you know we are using forecasting model so we must require weather data or time series so we started with weather data now. The main motive was to roll rain/snow with store and date level.

SELECT RH.storeNum as storeNum1, count(rh.ReceiptID) as count\_of\_ReceiptID, RAIN\_DATA.RAIN, rh.ReceiptDate  
FROM `driven-data-proto.teradata.ReceiptHeaders` rh  
INNER JOIN `driven-data-proto.teradata.ReceiptLineItems` li on rh.ReceiptID = li.ReceiptID  
INNER JOIN `driven-data-proto.teradata.SkuMap` sm On Concat(TRIM(li.ProductCode), '-', TRIM(li.Service)) = sm.ProductSKU  
Left join (select  
            STC.storenum  as STCstoreNum ,  
            STC.wx\_station as STCwx\_station,  
            w.PREC\_AWXC AS RAIN,  
            w.SNOW\_AWXC AS SNOW,  
            w.R\_DATE  as wR\_DATE  
            from `driven-teradata.t5analytics.Weather`  as w  
            JOIN  
            (SELECT storenum, wx\_station, s.zipcode as Szipcode#, count(\*)  
            FROM `driven-data-proto.teradata.Take5\_Shop\_Master` s  
            left join `driven-teradata.t5analytics.Zip\_Poatal\_Code` ws  
            on cast(s.zipcode as int64) = ws.zip\_code  
            group by 1,2,3  
            order by 3 desc ) as STC  
            ON STC.wx\_station= w.CODE  
            where w.R\_DATE  between '2018-01-01' AND '2018-09-20'  
            ) RAIN\_DATA  
ON (RH.storeNum = RAIN\_DATA.STCstoreNum AND rh.ReceiptDate = CAST(RAIN\_DATA.wR\_DATE AS STRING)  
)  
where RH.ReceiptDate  between '2018-01-01' AND '2018-09-20'  
AND RAIN\_DATA.RAIN  is NOT  NULL  
group by RH.storeNum, rh.ReceiptDate, RAIN\_DATA.RAIN

## Competitor level analysis

To find out the number of competitor we have used below query

### Direct Competitor with in 5 miles

SELECT T1.Shop\_Number as Shop\_Number

                            , Tm.StoreNum

                            , T1.Count\_DIRECT\_COMPETITOR as Count\_DIRECT\_COMPETITOR

                            FROM `driven-data-proto.teradata.Take5\_Shop\_Master`  TM

                            JOIN (

                                  select  CAST(Shop\_Number AS NUMERIC ) as Shop\_Number

                                  ,count( DIRECT\_COMPETITOR )  as Count\_DIRECT\_COMPETITOR

                                  FROM `driven-teradata.t5analytics.Direct\_Competitors`  IC

                                  where  DISTANCE\_\_MILES\_ <= 5

                                  group by Shop\_Number

                                  ) T1

                                  ON CAST(Shop\_Number AS NUMERIC) = CAST(TM.StoreNum  AS NUMERIC)

### Indirect Competitor with in 5 miles

SELECT T1.Shop\_Number as ShopeNumberIC

, Tm.StoreNum

                                  , T1.Count\_INDIRECT\_COMPETITOR as Count\_INDIRECT\_COMPETITOR

                                  FROM `driven-data-proto.teradata.Take5\_Shop\_Master`  TM

                                  JOIN (

                                        select  CAST(Shop\_Number AS NUMERIC ) as Shop\_Number

                                        ,count(INDIRECT\_COMPETITOR)  as Count\_INDIRECT\_COMPETITOR

                                        FROM `driven-teradata.t5analytics.Indirect\_Competitors`  IC

                                        where  DISTANCE\_\_MILES\_ <=5

                                        group by Shop\_Number

                                        ) T1

                                        ON CAST(Shop\_Number AS NUMERIC) = CAST(TM.StoreNum  AS NUMERIC)

## Vehicles data

When we have competitor we need to check how many vehicle are register within 5 ,2 miles.

### Vehicles /Car count within 5 miles of our stores

CREATE TEMPORARY FUNCTION Degrees(radians FLOAT64) RETURNS FLOAT64 AS  
(  
  (radians\*180)/(acos(-1))  
);

 CREATE TEMPORARY FUNCTION Radians(degrees FLOAT64) AS (  
  (degrees\*(acos(-1)))/180  
);

CREATE TEMPORARY FUNCTION HavDistanceMi (lat FLOAT64, lon FLOAT64, lat1 FLOAT64, lon1 FLOAT64) AS (  
     Degrees(   
      ACOS(   
        COS( Radians(lat1) ) \*   
        COS( Radians(lat) ) \*    
        COS( Radians(lon1 ) -    
        Radians( lon ) ) +    
        SIN( Radians(lat1) ) \*    
        SIN( Radians( lat ) )   
        )   
    ) \* 69.0  
);

select StoreNum, sum(sumOfTotal) as NearbyCars   
from ( select  BASE.StoreNum,BASE.StoreZipCode ,BASE.distance\_in\_mi ,BASE.ZIP\_CODE ,VIO.VEHICLE\_TYPE , sum(Total) sumOFTotal  
, COUNT(VIO.VEHICLE\_TYPE) countOFVEHICLE\_TYPE  
from ( select a.\* from (  
SELECT (lpad(License\_Number, 4, '0')) as  StoreNum, Location\_PostalCode as StoreZipCode,  LPAD(CAST(ZIPS.ZIP\_CODE AS STRING), 5, '0') ZIP\_CODE,  
  HavDistanceMi(safe\_cast(TSM.Location\_Latitude as float64),   
  safe\_cast(TSM.Location\_Longitude as float64),  
  ZIPS.LAT, ZIPS.LONG) as distance\_in\_mi  
from `driven-data-proto.teradata.DrivenBrandsShopMaster` TSM,  
`driven-teradata.t5analytics.Zip\_Poatal\_Code` ZIPS  
where abs(safe\_cast(TSM.Location\_Latitude as float64) - ZIPS.LAT) <= .2   
and abs(safe\_cast(TSM.Location\_Longitude as float64) - ZIPS.LONG) <= .2  
and  alignment\_brand = 'Take 5'  
) a where a.distance\_in\_mi <= 5  
GROUP BY 1,2,3,4) BASE  
LEFT JOIN `driven-data-proto.teradata.201804\_VIO` vio  
on BASE.ZIP\_CODE = VIO.ZIP\_CODE  
where registration\_category in ('PERSONAL', 'SMALL FLEET')  
GROUP BY 1,2,3,4,5 ) carsbyzip group by 1 order by 1

## Market stores saturation

Now we want to see how many are nearby store and how we can assign a particular number of car to each store.

select carNearby.\*  
,Innerdata.ShopeNumber  
,Innerdata.Count\_DIRECT\_COMPETITOR  
,Innerdata.Count\_INDIRECT\_COMPETITOR  
,(carNearby.NearbyCars/(Innerdata.Count\_INDIRECT\_COMPETITOR + Innerdata.Count\_DIRECT\_COMPETITOR + 1 ) ) as stores\_saturation  
from `driven-teradata.t5analytics.Store\_NearbyCar` carNearby  
   JOIN (  
             Select DC.Count\_DIRECT\_COMPETITOR as Count\_DIRECT\_COMPETITOR  
             , IC.Count\_INDIRECT\_COMPETITOR as Count\_INDIRECT\_COMPETITOR  
             , CAST(dc.ShopeNumber as NUMERIC) as ShopeNumber  
             , CAST(IC.ShopeNumberIC as NUMERIC) as ShopeNumberIC  
             from  
                           (

                           SELECT T1.Shop\_Number as ShopeNumber  
                           , Tm.StoreNum  
                           , T1.Count\_DIRECT\_COMPETITOR as Count\_DIRECT\_COMPETITOR  
                           FROM `driven-data-proto.teradata.Take5\_Shop\_Master`  TM  
                           JOIN (  
                                 select  CAST(Shop\_Number AS NUMERIC ) as Shop\_Number  
                                 ,count( DIRECT\_COMPETITOR )  as Count\_DIRECT\_COMPETITOR  
                                 FROM `driven-teradata.t5analytics.Direct\_Competitors`  IC  
                                 #where  Shop\_Number= 258  
                                 group by Shop\_Number  
                                 ) T1  
                                 ON CAST(Shop\_Number AS NUMERIC) = CAST(TM.StoreNum  AS NUMERIC) ) DC  
             JOIN  
                                 (  
                                 SELECT T1.Shop\_Number as ShopeNumberIC  
                                 , Tm.StoreNum  
                                 , T1.Count\_INDIRECT\_COMPETITOR as Count\_INDIRECT\_COMPETITOR  
                                 FROM `driven-data-proto.teradata.Take5\_Shop\_Master`  TM  
                                 JOIN (  
                                       select  CAST(Shop\_Number AS NUMERIC ) as Shop\_Number  
                                       ,count(INDIRECT\_COMPETITOR)  as Count\_INDIRECT\_COMPETITOR  
                                       FROM `driven-teradata.t5analytics.Indirect\_Competitors`  IC  
                                       #where  Shop\_Number= 258  
                                       group by Shop\_Number  
                                       ) T1  
                                       ON CAST(Shop\_Number AS NUMERIC) = CAST(TM.StoreNum  AS NUMERIC) )IC

              ON ic.ShopeNumberIC = DC.ShopeNumber ) Innerdata  
ON CAST(Innerdata.ShopeNumber as NUMERIC) =CAST(carNearby.StoreNum as NUMERIC)

## Competitor pricing Information

select CBSATITLE.CBSA\_TITLE ,CBSATITLE.StoreNum   ,\_cohortamountformerchant\_  
      ,\_cohorttransactionsformerchant\_      ,merchant  
      ,CAST(\_cohort\_dpt\_formerchant\_ as STRING) \_cohort\_dpt\_formerchant\_   , month    
from `driven-teradata.t5analytics.Merchant\_Market\_size` as MMS   
JOIN (

  select CBSA\_TITLE ,newCBSA.StoreNum as StoreNum , count(\*)  from `driven-teradata.t5analytics.CBSA\_Code`    
        RIGHT JOIn (  
                SELECT StoreNum , ZipCode    
                , cbsa              FROM `driven-data-proto.teradata.Take5\_Shop\_Master`   
                JOIN  `driven-teradata.t5analytics.CBSA\_CODE\_NEW`   
                ON (ZipCode) =  CAST(ZIP AS STRING)  
                group by 1,2,3  ) as newCBSA  
        ON CBSA\_CODE=newCBSA.cbsa group by 1, 2   
        )  CBSATITLE   
 ON CBSATITLE.CBSA\_TITLE = MMS.state  
   
   
   
   
   
SELECT #substr(CAST(month as STRING), 1, 7)  as month1,  
month  
, StoreNum  
,CBSA\_TITLE  
,  
avg(CASE WHEN merchant = 'Jiffy Lube' THEN (CAST(SUBSTR(\_cohort\_dpt\_formerchant\_, STRPOS(\_cohort\_dpt\_formerchant\_, '$') + 1, LENGTH(\_cohort\_dpt\_formerchant\_)) as FLOAT64))     ELSE NULL END) Jiffy\_Lube,  
avg(CASE WHEN merchant = 'Grease Monkey' THEN (CAST(SUBSTR(\_cohort\_dpt\_formerchant\_, STRPOS(\_cohort\_dpt\_formerchant\_, '$') + 1, LENGTH(\_cohort\_dpt\_formerchant\_)) as FLOAT64))      ELSE NULL END) Grease\_Monkey,  
avg(CASE WHEN merchant = 'Take 5 Oil Change' THEN (CAST(SUBSTR(\_cohort\_dpt\_formerchant\_, STRPOS(\_cohort\_dpt\_formerchant\_, '$') + 1, LENGTH(\_cohort\_dpt\_formerchant\_)) as FLOAT64))      ELSE NULL END) Take\_5\_Oil\_Change,  
avg(CASE WHEN merchant = 'Express Oil Change' THEN (CAST(SUBSTR(\_cohort\_dpt\_formerchant\_, STRPOS(\_cohort\_dpt\_formerchant\_, '$') + 1, LENGTH(\_cohort\_dpt\_formerchant\_)) as FLOAT64))      ELSE NULL END) Express\_Oil\_Change,  
avg(CASE WHEN merchant = 'Valvoline Express Care' THEN (CAST(SUBSTR(\_cohort\_dpt\_formerchant\_, STRPOS(\_cohort\_dpt\_formerchant\_, '$') + 1, LENGTH(\_cohort\_dpt\_formerchant\_)) as FLOAT64))      ELSE NULL END) Valvoline\_Express\_Care  
FROM `driven-teradata.t5analytics.Competitor\_pricing`   
GROUP BY 1,2 , 3  
order by month desc

## Holiday Data

To feed in model we have created a dummy holiday calendar table and we are using that juts to check do we have any impact on forecasting /optimization .

## Final View

After doing all the Analysis we have created a final view which has all the important features.

SELECT main.\* ,

ifnull(round((safe\_divide(GROSS\_PRICE, UNITS)-(safe\_divide(GROSS\_PRICE, UNITS)\*safe\_divide(APPLIED\_MKDOWN , SubTotal))),2),0) AS net\_price

,cal.IS\_Holiday as IS\_HOLIDAY, cal.IS\_HOLIDAY\_3DAYS, cal.IS\_HOLIDAY\_5DAYS from

(SELECT rh.TransactionID as TransactionID #1

,rh.ReceiptDate as DATE

,rh.SubTotal AS SubTotal

,RH.storeNum as SHOP #4

,0 AS LOST\_DEMAND

,9999 AS STOCK\_LEVEL

,ifnull(RAIN\_DATA.RAIN, 0) as RAIN

,ifnull(RAIN\_DATA.Snow, 0) as SNOW #8

,li.LineItemTotal AS GROSS\_PRICE

,ifnull(rd.discountamt, 0) as APPLIED\_MKDOWN

,li.TotalCOGS as LICOST #11

,li.UNITS AS UNITS

,li.ProductSubGroup as PRODUCT #13

,li.ProductGroup as PRODDUCTGROUP

,SC.cluster as CLUSTER

,MS.NearbyCars as NEARBYCAR

,MS.Count\_DIRECT\_COMPETITOR AS Count\_DIRECT\_COMPETITOR

,MS.Count\_INDIRECT\_COMPETITOR as Count\_INDIRECT\_COMPETITOR

,MS.stores\_saturation as STORE\_SATURATION

FROM `driven-data-proto.teradata.ReceiptHeaders` rh

LEFT JOIN `driven-data-proto.teradata.DrivenBrandsShopMaster` TSM

ON (lpad(License\_Number, 4, '0') =rh.storeNum)

INNER JOIN `driven-teradata.t5analytics.StoreClusters` SC

ON SC.storenum = lpad(License\_Number, 4, '0')

################ START SATURATION

LEFT JOIN

(SELECT carNearby.StoreNum as StoreNum

,carNearby.NearbyCars as NearbyCars

,Innerdata.ShopeNumber as ShopeNumber

,Innerdata.Count\_DIRECT\_COMPETITOR as Count\_DIRECT\_COMPETITOR

,Innerdata.Count\_INDIRECT\_COMPETITOR as Count\_INDIRECT\_COMPETITOR

,(carNearby.NearbyCars/(Innerdata.Count\_INDIRECT\_COMPETITOR + Innerdata.Count\_DIRECT\_COMPETITOR + 1 ) ) as stores\_saturation

FROM `driven-teradata.t5analytics.Store\_NearbyCar\_With\_newTable` carNearby

JOIN (Select DC.Count\_DIRECT\_COMPETITOR as Count\_DIRECT\_COMPETITOR

, IC.Count\_INDIRECT\_COMPETITOR as Count\_INDIRECT\_COMPETITOR

, CAST(dc.ShopeNumber as NUMERIC) as ShopeNumber

, CAST(IC.ShopeNumberIC as NUMERIC) as ShopeNumberIC

FROM

(SELECT T1.Shop\_Number as ShopeNumber

, lpad(License\_Number, 4, '0')

, T1.Count\_DIRECT\_COMPETITOR as Count\_DIRECT\_COMPETITOR

FROM `driven-data-proto.teradata.DrivenBrandsShopMaster` TM

JOIN (select CAST(Shop\_Number AS NUMERIC ) as Shop\_Number

,count( DIRECT\_COMPETITOR ) as Count\_DIRECT\_COMPETITOR

FROM `driven-teradata.t5analytics.Direct\_Competitors` IC

group by Shop\_Number

) T1

ON CAST(Shop\_Number AS NUMERIC) = CAST(lpad(License\_Number, 4, '0') AS NUMERIC)

where alignment\_brand = 'Take 5'

) DC

JOIN (SELECT T1.Shop\_Number as ShopeNumberIC

, lpad(License\_Number, 4, '0')

, T1.Count\_INDIRECT\_COMPETITOR as Count\_INDIRECT\_COMPETITOR

FROM `driven-data-proto.teradata.DrivenBrandsShopMaster` TM

JOIN (SELECT CAST(Shop\_Number AS NUMERIC ) as Shop\_Number

,count(INDIRECT\_COMPETITOR) as Count\_INDIRECT\_COMPETITOR

FROM `driven-teradata.t5analytics.Indirect\_Competitors` IC

GROUP BY Shop\_Number

) T1

ON CAST(Shop\_Number AS NUMERIC) = CAST(lpad(License\_Number, 4, '0') as NUMERIC)

WHERE alignment\_brand = 'Take 5'

) IC

ON IC.ShopeNumberIC = DC.ShopeNumber ) Innerdata

ON CAST(Innerdata.ShopeNumber as NUMERIC) =CAST(carNearby.StoreNum as NUMERIC)

) as MS

ON ( MS.StoreNum = lpad(License\_Number, 4, '0') )

################ END SATURATION

INNER JOIN

(SELECT case when s1.mastergroup like '%DIESEL%' then concat('DIESEL ', s2.productsubgroup) else s2.productsubgroup end as ProductSubGroup

,li.TransactionID as TransactionID

,li.ProductCode as ProductCode

,li.Service as Service

,s2.ProductGroup as ProductGroup

, sum(TotalCOGS) as TotalCOGS

, round(sum(li.LineItemTotal),2) as LineItemTotal

, count(\*) as UNITS

FROM (SELECT TransactionID as TransactionID

,ProductCode as ProductCode

,Service as Service

,LineItemTotal as LineItemTotal

,TotalCOGS as TotalCOGS

,round(LineItemTotal - TotalCOGS,2) as GrossProfit

,round(SAFE\_DIVIDE((LineItemTotal - TotalCOGS), LineItemTotal),2) as GrossMargin

FROM (SELECT \*

,round(( PartsCOGS + LubricantCOGS),2) as TotalCOGS

FROM`driven-data-proto.teradata.ReceiptLineItems`

)) li

LEFT JOIN `driven-data-proto.teradata.Take5\_ProductSKUTest` s1

On Concat(TRIM(li.ProductCode), '-', TRIM(li.Service)) = s1.ProductSKU

INNER JOIN `driven-data-proto.teradata.SkuMap` s2

On Concat(TRIM(li.ProductCode), '-', TRIM(li.Service)) = s2.ProductSKU

WHERE s2.productgroup in ('OIL CHANGE', 'BIG4')

GROUP BY 1,2,3,4,5

) li

ON rh.TransactionID = li.TransactionID

LEFT JOIN (SELECT TransactionID , SUM(discountamt) as discountamt

FROM `driven-data-proto.teradata.ReceiptDiscounts`

group by 1

)rd

on li.TransactionID = rd.TransactionID

LEFT JOIN (SELECT

STC.storenum as STCstoreNum ,

STC.wx\_station as STCwx\_station,

w.PREC\_AWXC AS RAIN,

w.SNOW\_AWXC AS SNOW,

w.R\_DATE as wR\_DATE

from `driven-teradata.t5analytics.Weather` as w

JOIN

(SELECT lpad(License\_Number, 4, '0') as storenum

, wx\_station

, cast(s.Location\_PostalCode as int64) as Szipcode

FROM `driven-data-proto.teradata.DrivenBrandsShopMaster` s

left join `driven-teradata.t5analytics.Zip\_Poatal\_Code` ws

on cast(s.Location\_PostalCode as int64) = ws.zip\_code

where alignment\_brand = 'Take 5'

group by 1,2,3

order by 3 desc ) as STC

ON STC.wx\_station= w.CODE

where w.R\_DATE > '2018-01-01'

) RAIN\_DATA

ON (RH.storeNum = RAIN\_DATA.STCstoreNum AND rh.ReceiptDate = CAST(RAIN\_DATA.wR\_DATE AS STRING)

)

where RH.ReceiptDate > '2018-01-01'

AND alignment\_brand = 'Take 5'

) main

Left JOIN `driven-teradata.t5analytics.Calendar` cal

on cast(main.DATE as date)= cast(cal.Date as date)

# Importing required data/Data Frame.

## Import data in import data module tab

Go to directory work/srv/dev/tab and open tab “import\_data\_module.R”, we are using this tab to import all the data and converting in data processing workflow.

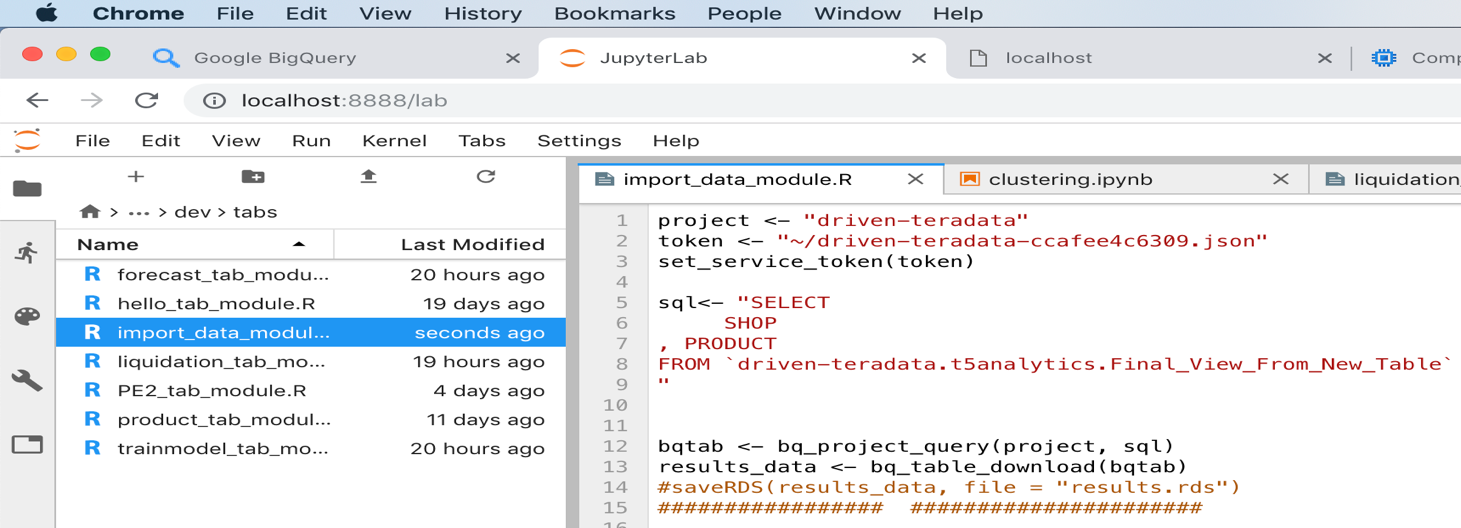
1. We need to connect with big query by running below command.
   * + 1. project <- "driven-teradata"
       2. token <- "~/driven-teradata-ccafee4c6309.json" (file should be present in same directory)
       3. set\_service\_token(token)
2. Example how to import data.

Let’s take an example of query (SELECT SHOP , PRODUCT FROM `driven-teradata.t5analytics.Final\_View\_From\_New\_Table` ).

Assuming, you had run above query without any issue. If yes then execute below command line by line.

* + 1. iproject <- "driven-teradata".
    2. token <- "~/driven-teradata-ccafee4c6309.json".
    3. set\_service\_token(token).
    4. sql <- “SELECT SHOP, PRODUCT FROM `driven-teradata.t5analytics.Final\_View\_From\_New\_Table`”
    5. bqtab <- bq\_project\_query(project, sql).
    6. results\_data <- bq\_table\_download(bqtab)

now in the variable results\_data has all the variable which you can use for your model training. below is the example of data importing.

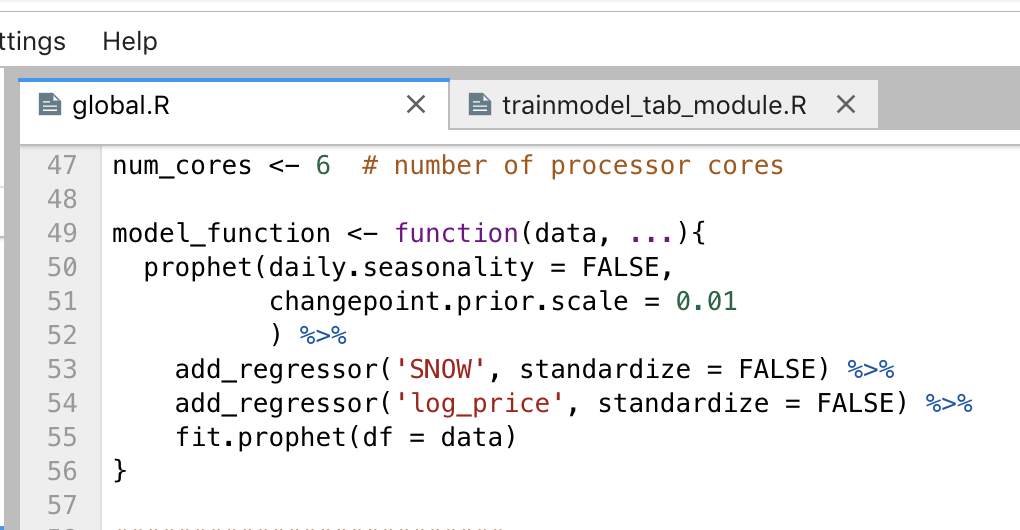


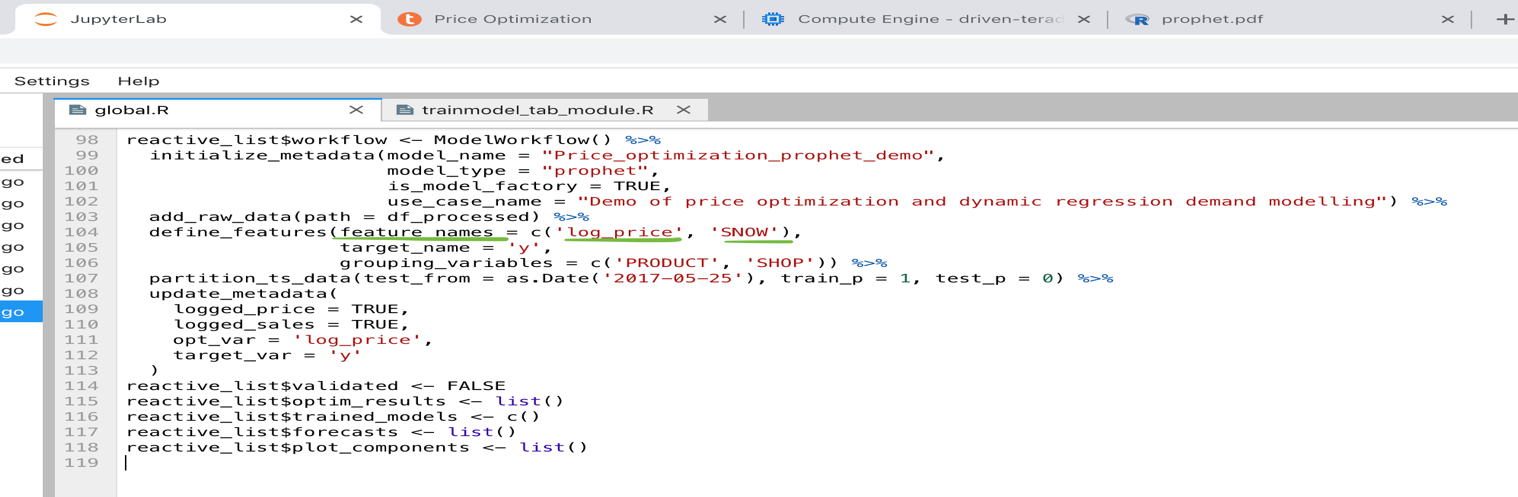
# Add\_regressor/feature to fit the model

The data frame passed to ‘fit‘ and ‘predict‘ will have a column with the specified name to be used as a regressor. When standardize = ’auto’, the regressor will be standardized unless it is binary. The regression coefficient is given a prior with the specified scale parameter. Decreasing the prior scale will add additional regularization.

## Current regressor.

* 1. Current regressor/feature present.
     1. To check how many regressor are currently present we need to go to directory “/work/dev/” then we will open file “global.R” and check below function





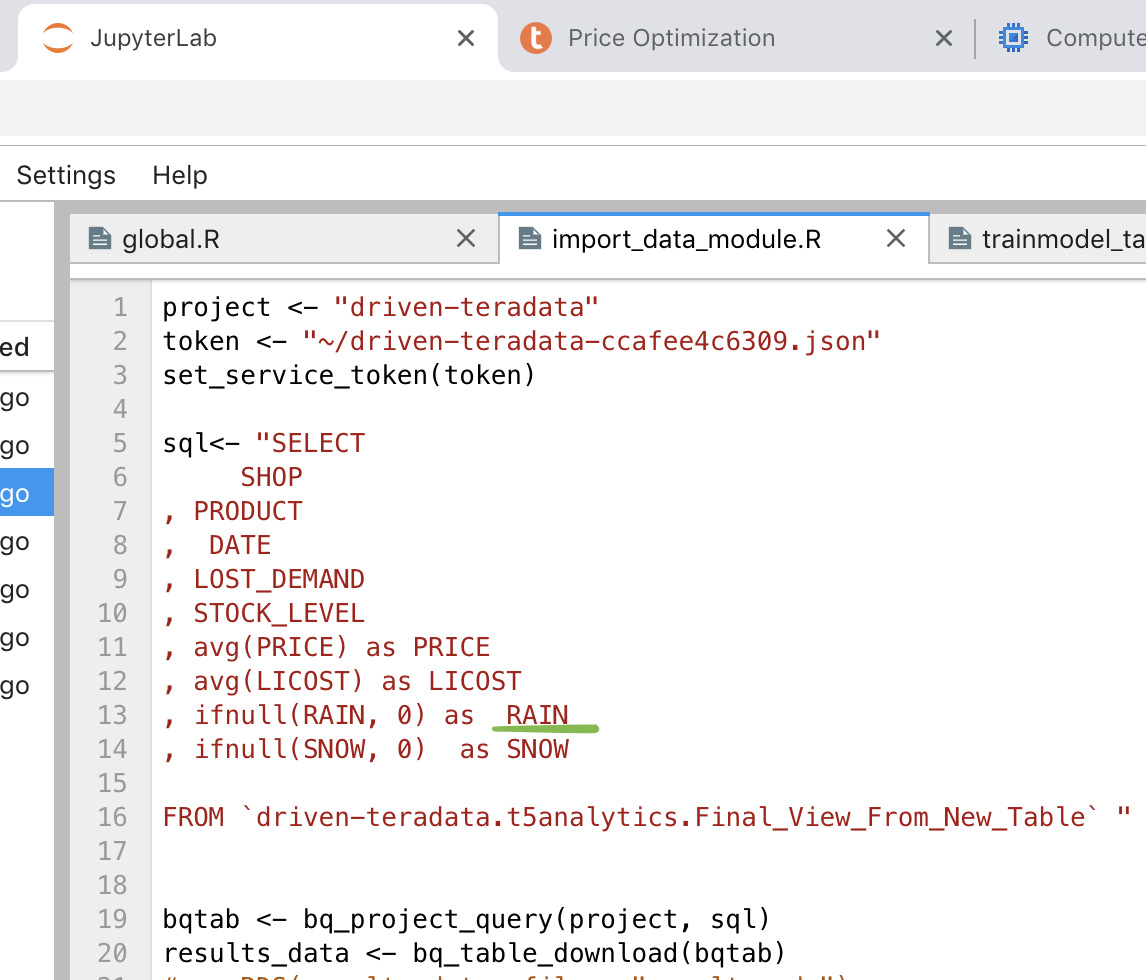
currently we are using only two regressor/feature to fit model

## Adding new regressor/feature

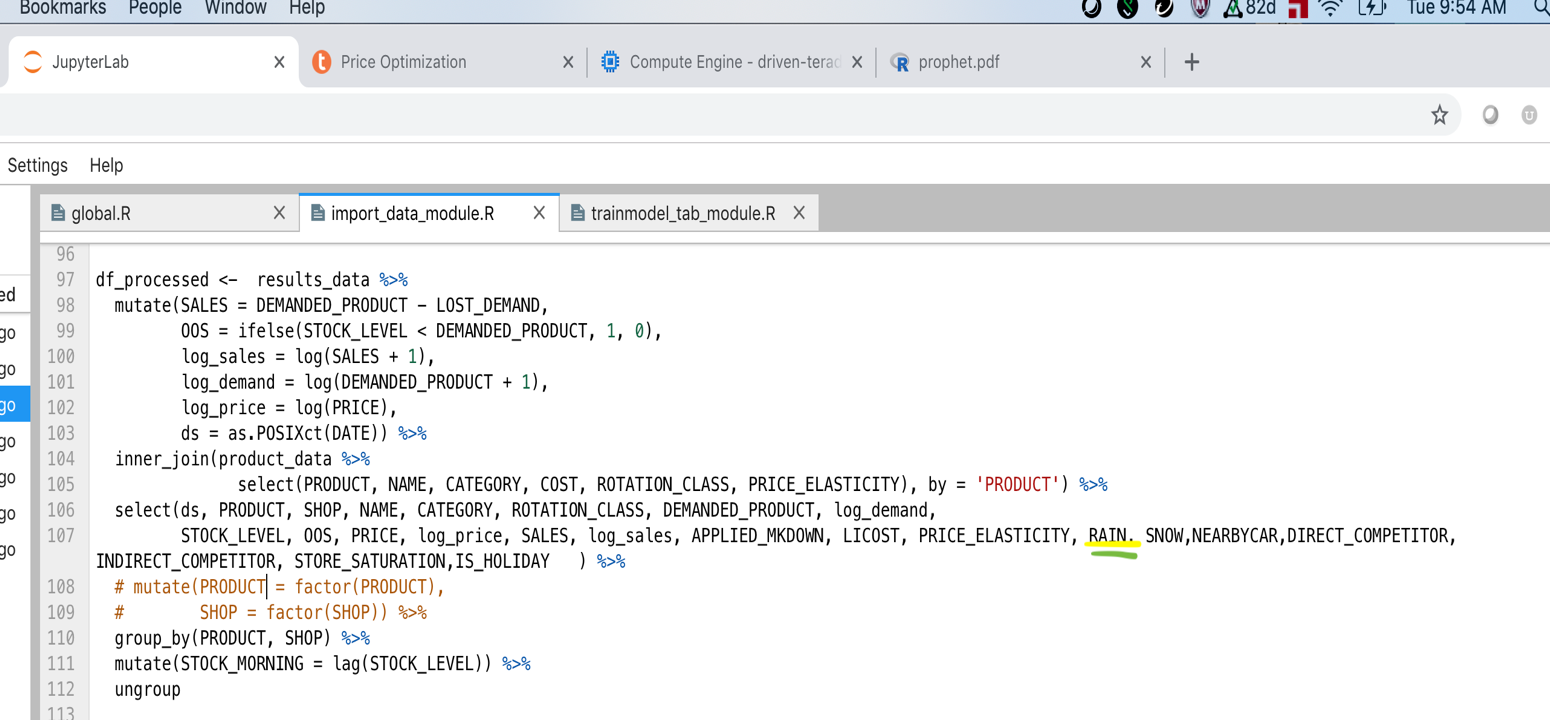
To add new feature we need to modify below tabs. For Demo purpose I am going to add RAIN as new regressor.

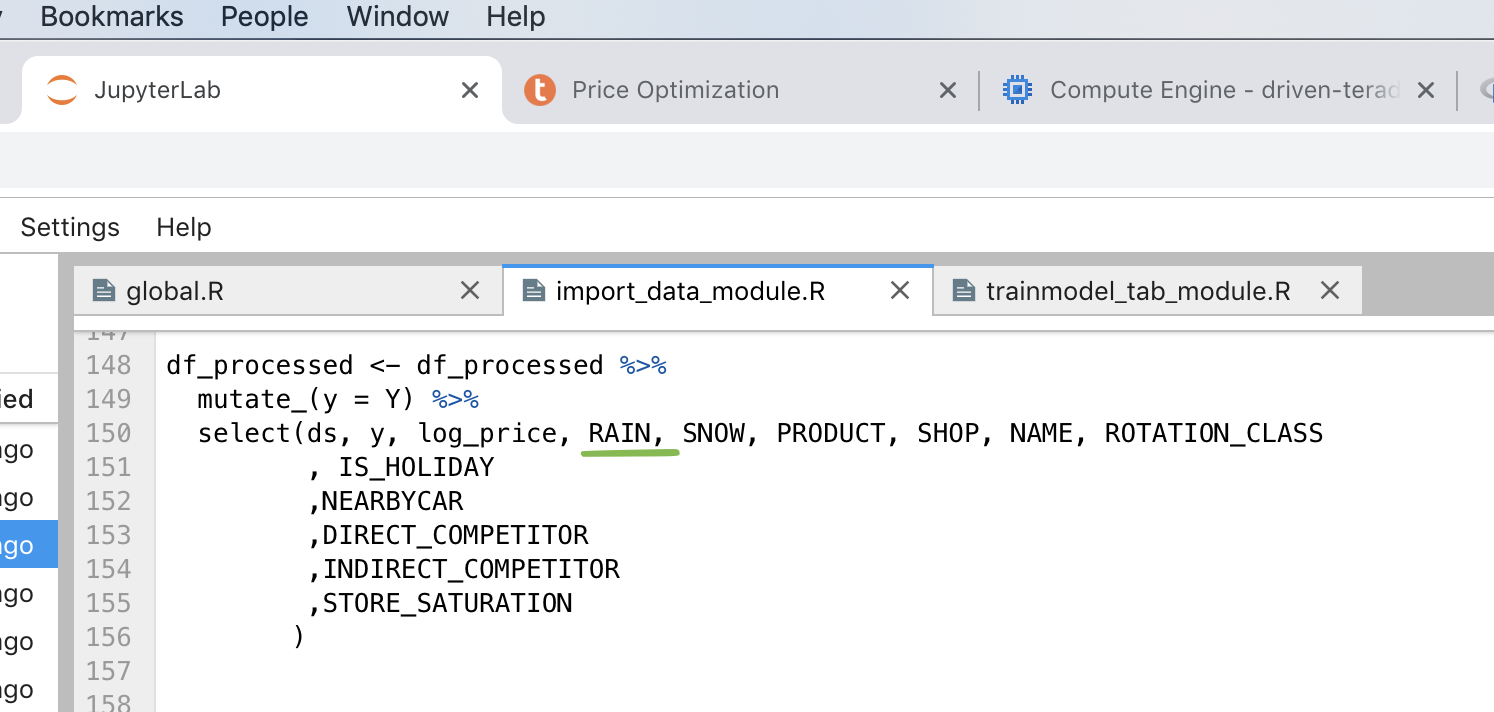
* + 1. Open the tab “import\_data\_module.R” and follow the below changes.

We need to make sure we add new feature in our data frame variable “*results\_data*” in tab

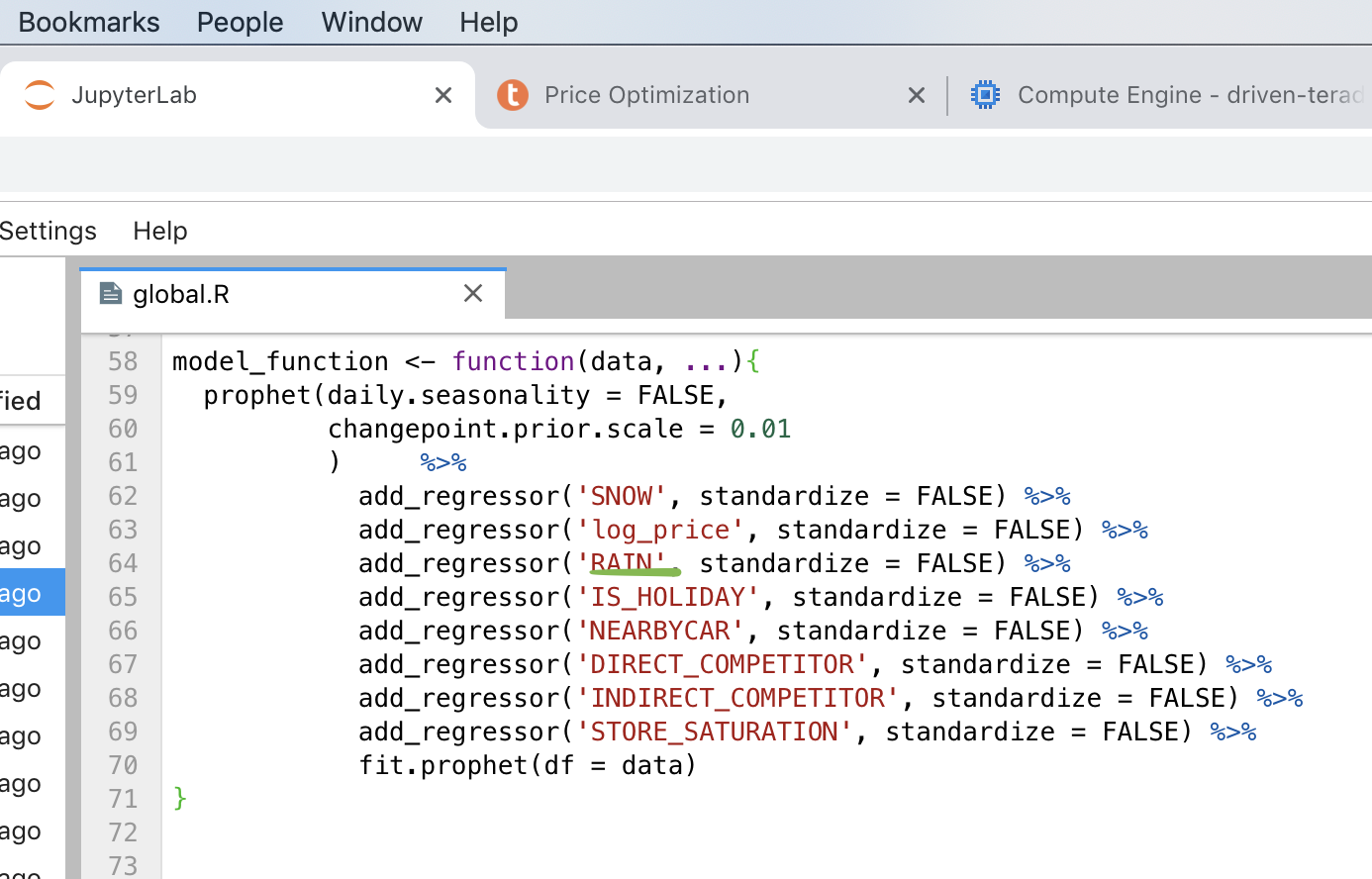


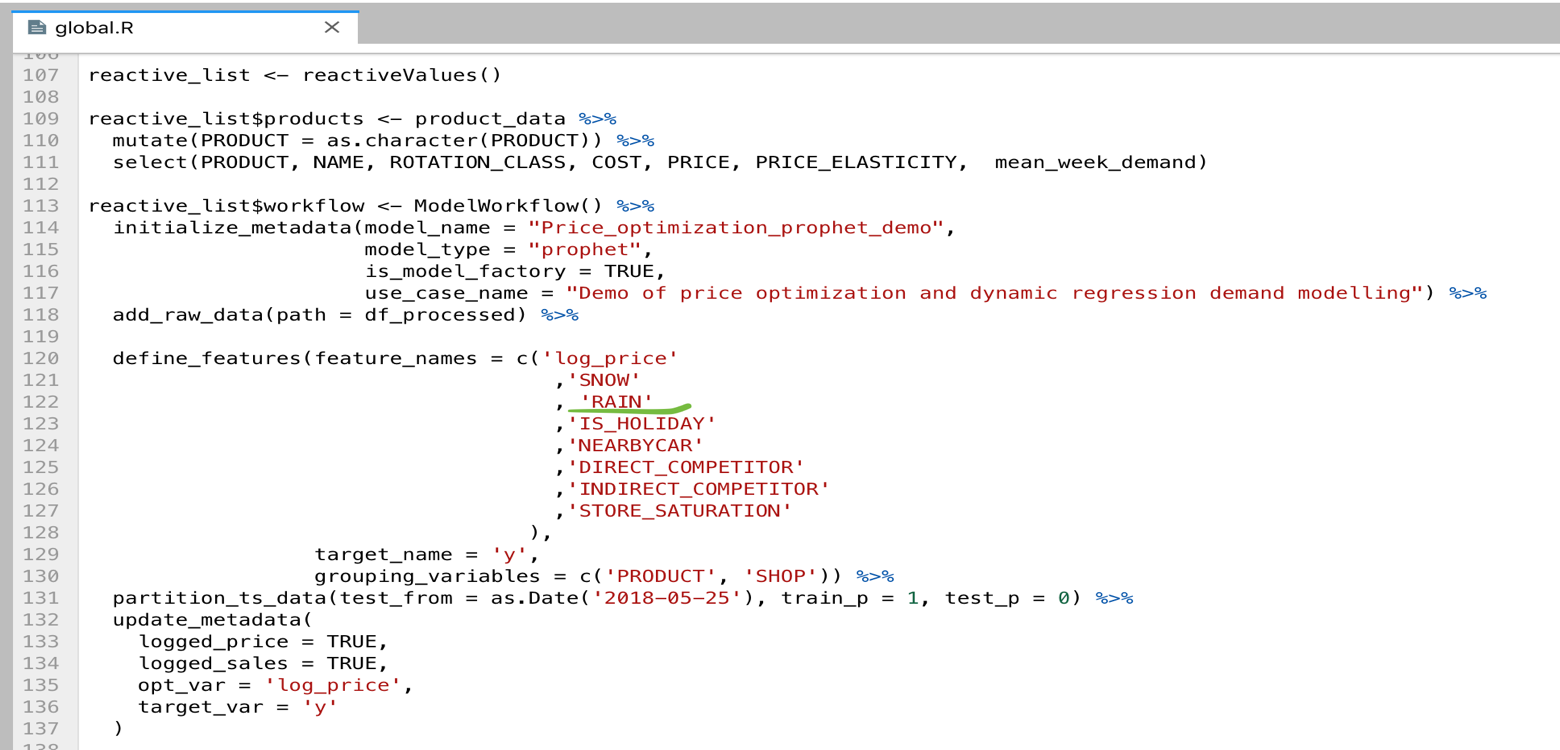
Then we need to add regressor to our data frame variable (df\_processed) by doing some modification/joining data like below.





* + 1. Now go to tab “global.R” add RAIN feature in function model\_function and also add in Feature\_Name variable like Below.

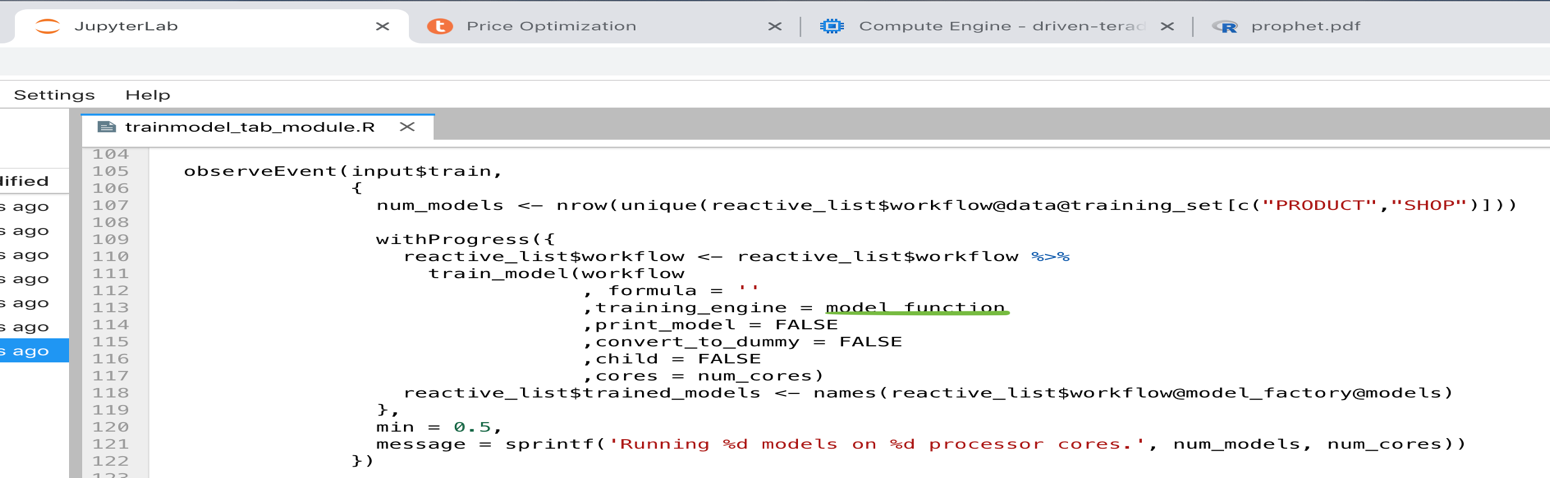




Here you can see we have added RAIN as regressor in feature list.

# Validating regressor in model training.

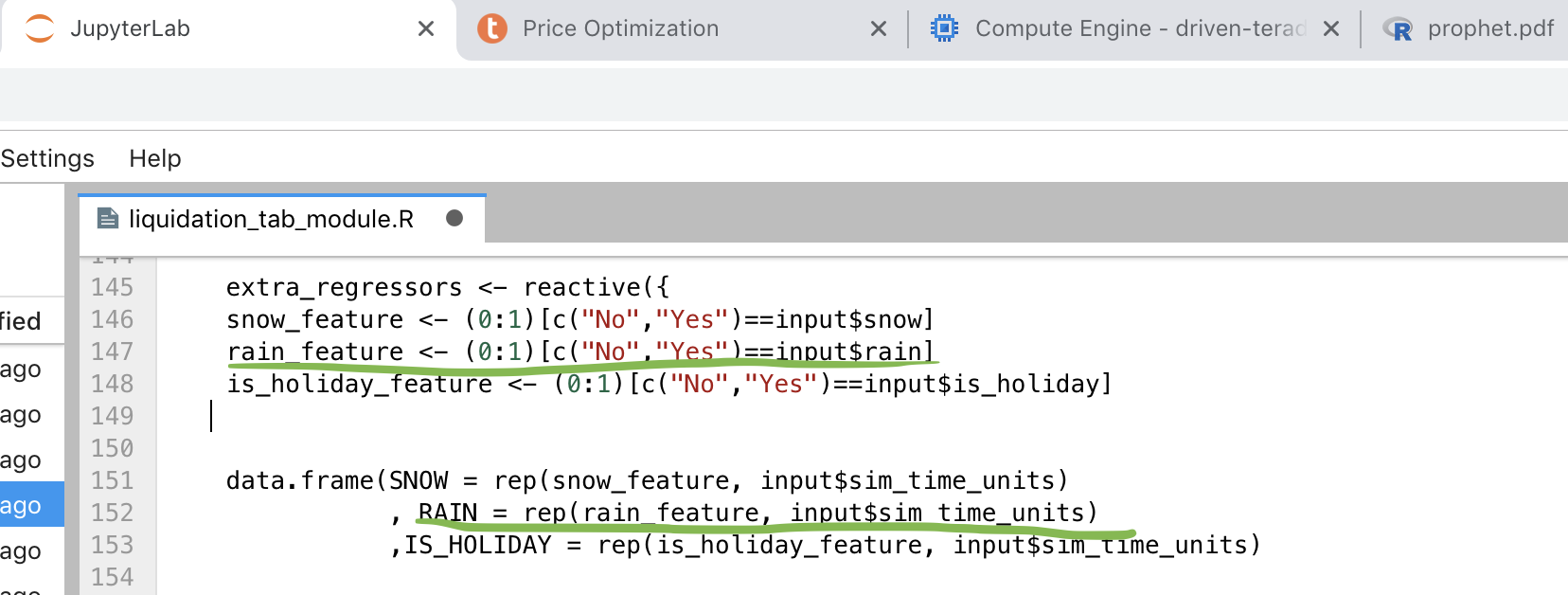
1. Now we need to open tab “trainmodel\_tab\_module.R” and need to validate are we passing function “model\_function” in the model training like below and you can validate regressor by printing the model training logs.



1. We need to add regressor to our forecasting tab (forecast\_tab\_module.R) in variable “feature\_list” as below.



1. Let’s jump to liquidation\_tab\_module.R tab , if you want to consider newly added feature for price optimization. You just need to modify below section of code.



# Important Query

For Store cluster on graph

SELECT storenum, CN.cluster as cluster, Location\_Latitude, Location\_Longitude , clustername

FROM `driven-teradata.t5analytics.StoreClusters` SC

JOIN `driven-data-proto.teradata.DrivenBrandsShopMaster` SM

ON SC.storenum = lpad( License\_Number,4,'0')

LEFT JOIN `driven-teradata.t5analytics.ClusterNames` CN

ON CN.cluster = SC.cluster

where Alignment\_Brand = 'Take 5'

For Competitor Data

select storeNum as SHOP,

CASE WHEN TRIM(Product) = 'Air Filters' Then 'AIR FILTER'

WHEN TRIM(Product) = 'Cabin Air Filters' then 'CABIN FILTER'

WHEN TRIM(Product) = 'Synthetic Blend (Oil Change-Gas)' then 'SYNBLEND'

WHEN TRIM(Product) = 'Conventional (Oil Change-Gas)' then 'CONVENTIONAL'

WHEN TRIM(Product) = 'Conventional Wiper Blades' then 'WIPER BLADES'

WHEN TRIM(Product) = 'Full Synthetic (Oil Change-Gas)' then 'HIGHMILE'

WHEN TRIM(Product) = 'Advanced Synthetic (Oil Change-Gas)' then 'SYNTHETHIC'

ELSE NUll END as PRODUCT,

CompetitorAvgPrice as maxPrice

from `driven-teradata.t5analytics.max\_price`

For store Address date

SELECT lpad(License\_Number, 4, '0') as License\_Number, Center\_Email2, Location\_Address ,Location\_City , Location\_State FROM `driven-data-proto.teradata.DrivenBrandsShopMaster` where alignment\_brand = 'Take 5'

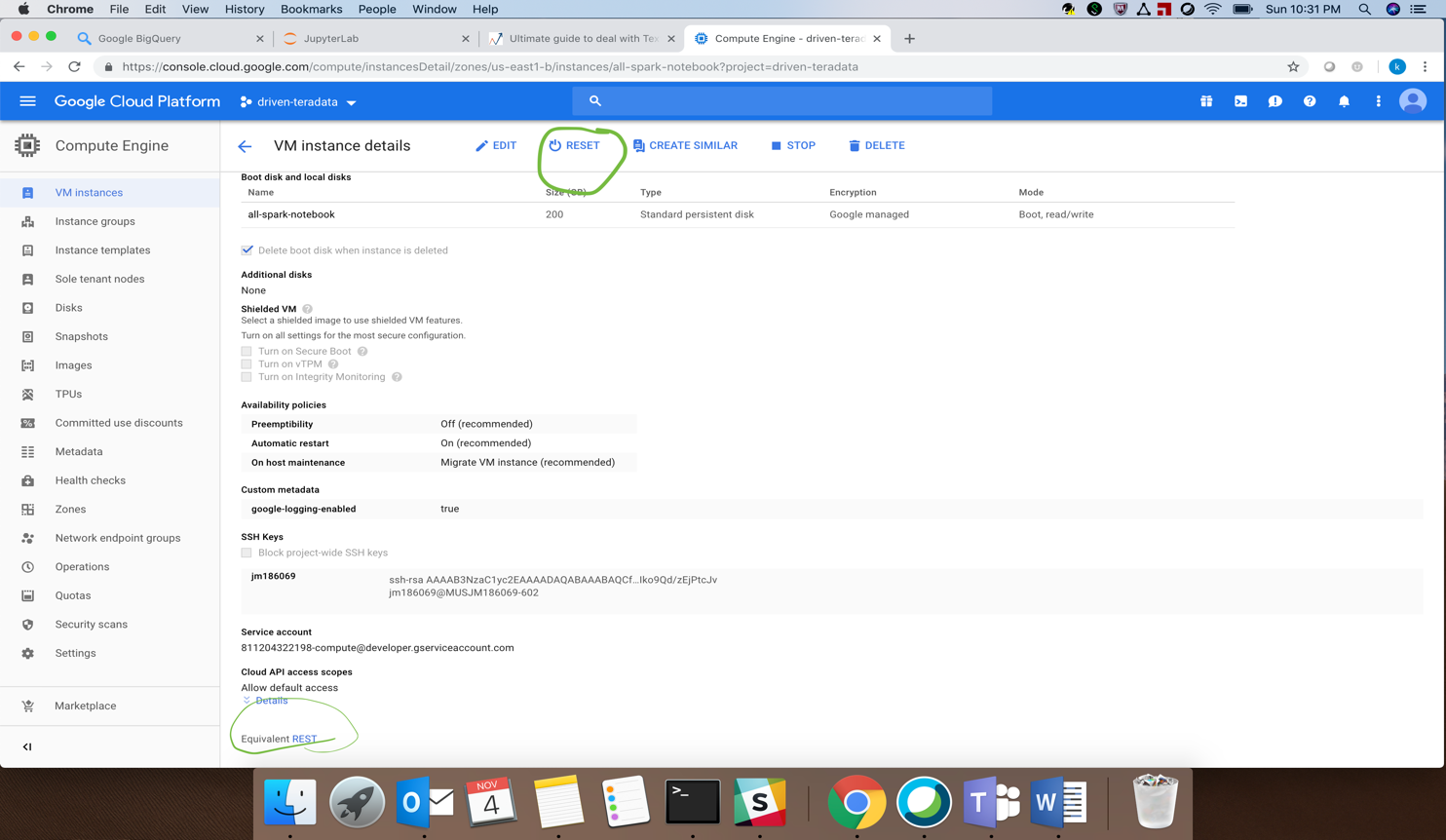
# Import Tables (Do not Delete )

* + - * BadStoreNum
      * AvgDailySalesByCat
      * Direct\_Competitors
      * Final\_Competitor\_Info
      * Store\_NearbyCar\_With\_newTable
      * StoreClusters
      * Sum\_Trans\_View
      * STORE\_NEAR\_ZIP\_CODE

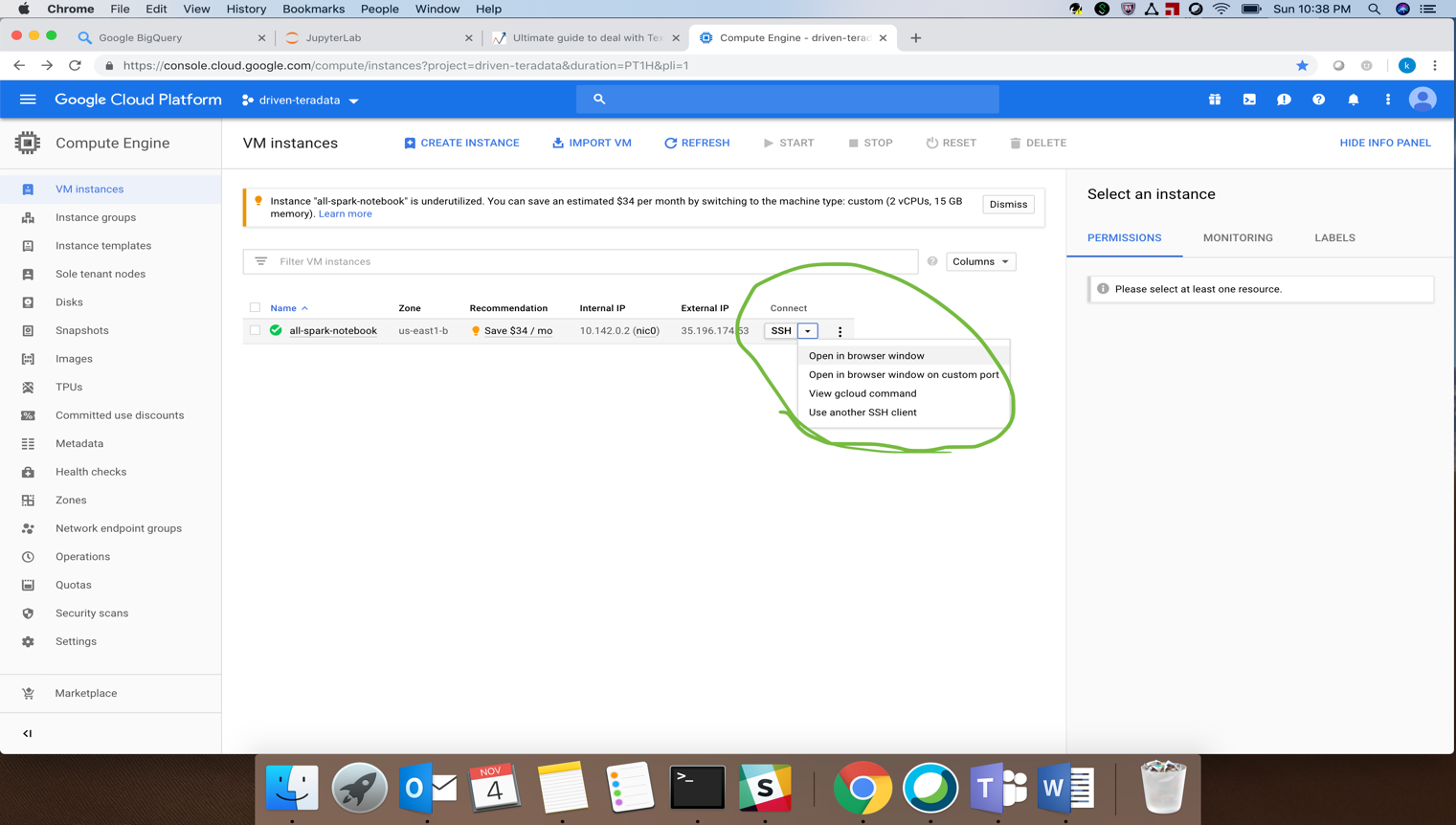
# How to Restart google server

Go to link <https://console.cloud.google.com/compute/instancesDetail/zones/us-east1-b/instances/all-spark-notebook?project=driven-teradata>

You will be able to find below screen and click in “RESET”



Now you can login to google server and use below command



Now you will be able to see below screen , where you run below command

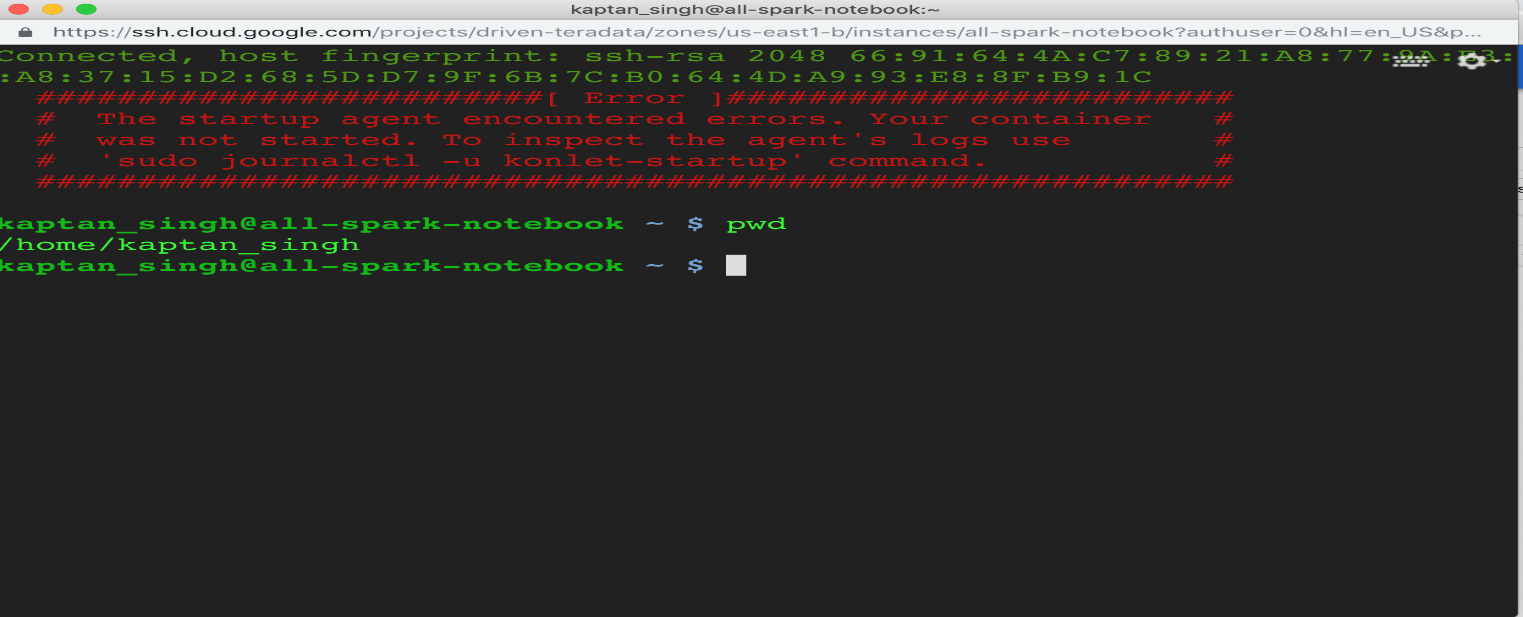
docker run --user root -e GRANT\_SUDO=yes -it --rm -p 8888:8888 -v /home/jack\_mccush/test:/home/jovyan/work jupyter/all-spark-notebook

docker run -d -p 3838:3838 -v /home/jack\_mccush/test/srv:/srv/shiny-server quantumobject/docker-shiny

export GOOGLE\_APPLICATION\_CREDENTIALS="/home/jack\_mccush/test/driven-teradata-ccafee4c6309.json"

scp -r -i ~/.ssh/google\_compute\_engine /Users/jm186069/Documents/GitHub/demand\_forecasting jm186069@35.196.174.53:/home/jack\_mccush/test/

docker run --rm -d -it -p 3838:3838 -v /home/jack\_mccush/test/srv:/srv/shiny-server dmmfdemo



Note :- you can take snapshot of the old container.

