Problem Statement:

Lisa, who is a Life sciences marketer conducts Marketing initiatives to engage Health Care Professionals (HCPs) mainly Doctors, to influence them for writing Rx (prescription) of her Pharma Products (MyProd1 and MyProd2). Lisa has received the attached file (data.csv) as her Target List (TL). She has only limited marketing budget, to help her maximize her returns she has approached us to help her find the high value doctors. It would greatly help her if she is able to segment the TL into 4 segments – Super High, High, Medium and Low value, on the basis of their likelihood of prescribing MyProd1 or MyProd2 in future.

Our Job is to Segment the TL into the 4 segments.

So we will be using clustering models to divide the dataset in to 4 segments.

Dataset:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 77247 entries, 0 to 77246
Data columns (total 23 columns):
State                         34969 non-null float64
Region                        34973 non-null float64
Division                      34973 non-null float64
Group_Name                    34977 non-null float64
MSA_Population_Size           34977 non-null float64
GENDER_CODE                   34949 non-null float64
PRESUMED_DEAD_FLAG            77247 non-null int8
Primary_TOP                   34977 non-null float64
Primary_PE                    34977 non-null float64
Primary_AD                    34977 non-null float64
Secondary                     34977 non-null float64
RX_Restriction_Indicator      77247 non-null int8
customer_id                   77247 non-null int64
MyProd1_Rx                    77247 non-null float64
MyProd2_Rx                    77247 non-null float64
CompProd1_Rx                  77247 non-null float64
CompProd2_Rx                  77247 non-null float64
CompProd3_Rx                  77247 non-null float64
Age                           34977 non-null float64
TOP                           77247 non-null int64
PE                            77247 non-null int64
SPECIAlITY                    77247 non-null int8
YrsPractice                   34977 non-null float64
dtypes: float64(17), int64(3), int8(3)
memory usage: 12.0 MB
```

We can see that dataset has 77247 rows and 23 columns.

Missing Value Analysis:

variables	missing percent
PRESUMED_DEAD_FLAG	100.00
RX_Restriction_Indicator	97.25
GENDER_CODE	54.76
State	54.73
Region	54.73
Division	54.73
ID	54.72
Primary_AD	54.72
Age	54.72
YrsPractice	54.72
Secondary	54.72
Primary_PE	54.72
Primary_TOP	54.72
MSA_Population_Size	54.72
Group_Name	54.72
PE	0.00
Ignore5	0.00
Ignore4	0.00
Ignore3	0.00
Ignore2	0.00
Ignore1	0.00
SPECIAlITY	0.00
MyProd1_Rx	0.00
TOP	0.00
CompProd3_Rx	0.00

Missing values more than 30 percent is not accepted. We have to delete the variable. But in this case, if we delete the important variables which describes the demographic information of the customer we may not able to segment the data properly. So Missing values are imputed with appropriate methods.

For column, "PRESUMED_DEAD_FLAG" columns has only two values as "D" and remaining are missing so we have to replace Not Dead with "N" or 0 and Dead with "Y" or 1.

For Column, RX_Restriction_Indicator negative flags are missing so we have imputed with "N" and "Y"

For Remaining values we have imputed with different methods.

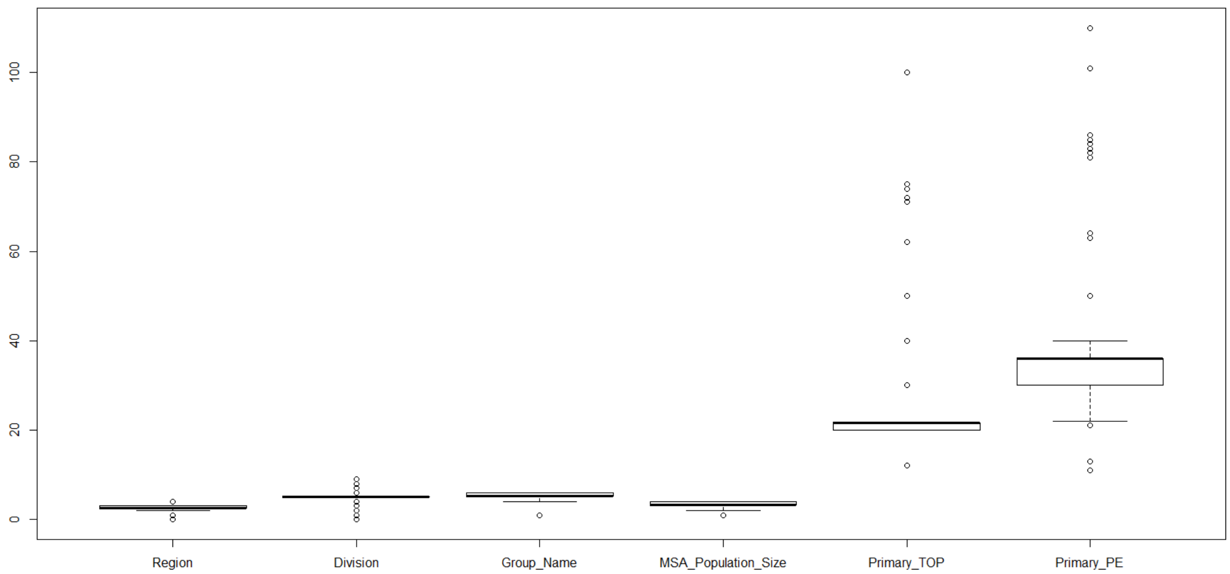
For Categorical: Imputation with Mode

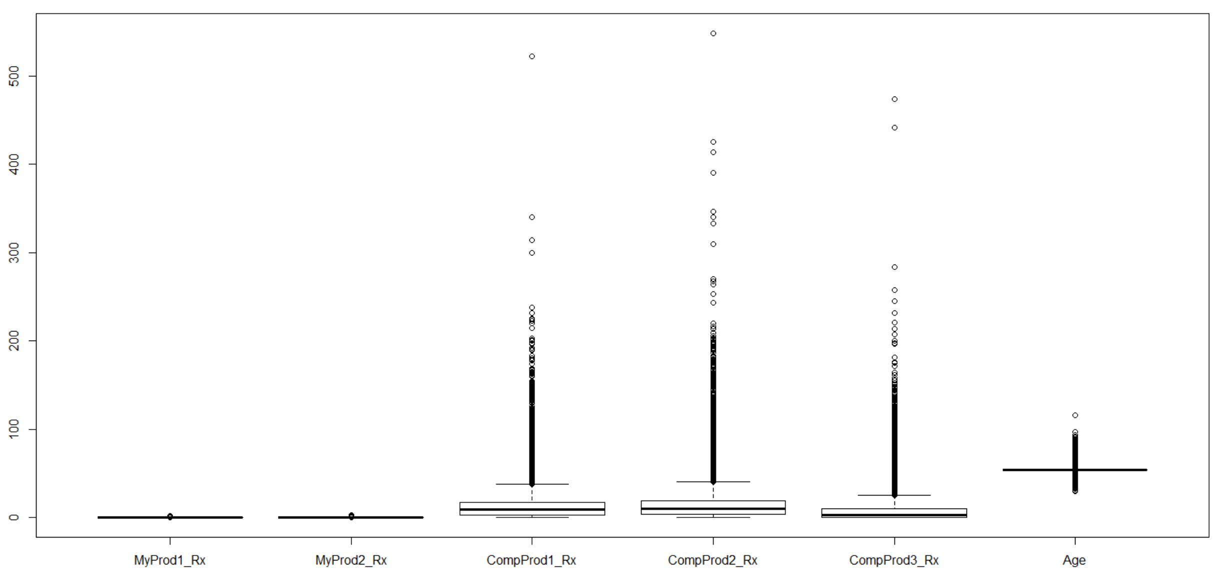
For Numerical: Imputation with Mean

In Python, we have implemented with KNNImputation which is taking lot of time to impute. So we have divided the data in to four splits and applied KNN imputation to avoid memory error.

Outlier Analysis:

Let us check the outliers in the dataset





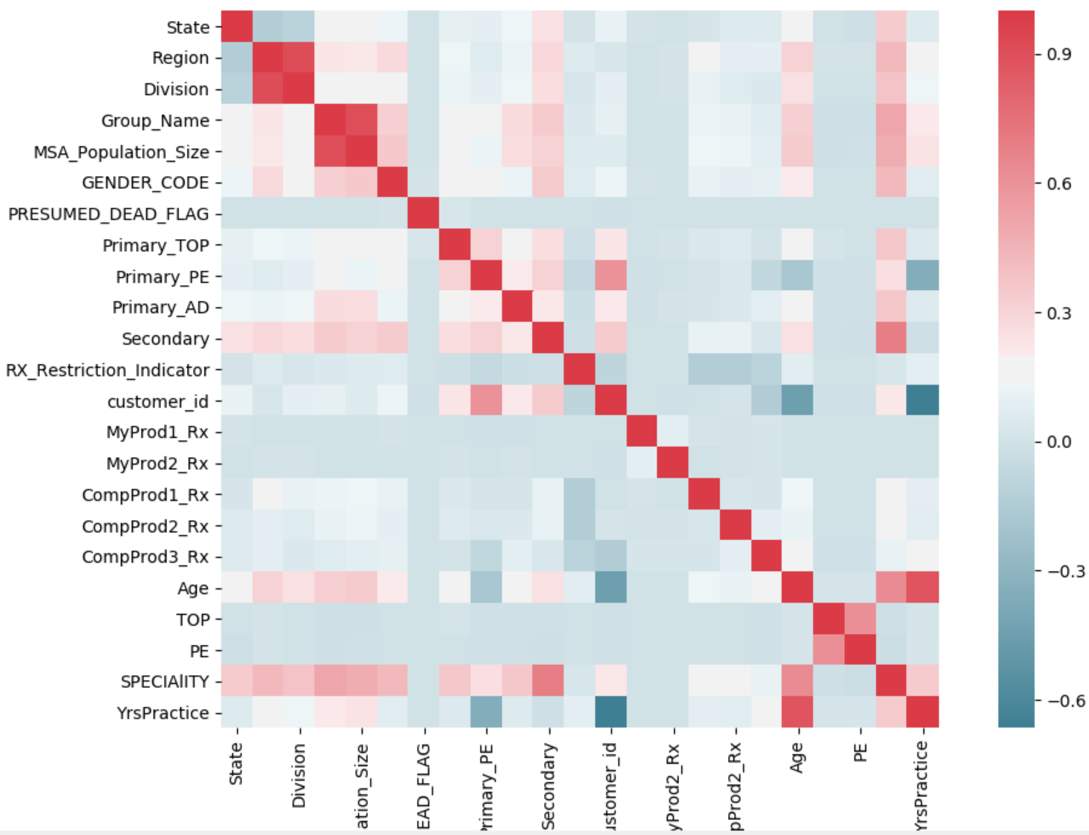
We can see that there are so many outliers present in each of the continuous variables.

So we have replaced these outliers with “na”

And these “na”’s are Imputed with mean method. In Python, we have used KNNImputation

Feature Selection:

Let us understand the relation between the different variables with the help of correlation plot.



Highly Correlated Variables:

Division and Region –Same Information(correlation 0.9)

Group_Name and MSA_Population_Size-Same Information(Correlation 0.9)

Age and YrsPractice –Same Information(Correlation 0.88)

"CompProd3_Rx" is of no use since we are dealing with prod1 and prod2.

"Ignore1","Ignore2","Ignore3","Ignore4","Ignore5","Ignore6","ID","ID2".

These are the variables which are of no use in segmenting the customers

So we are deleting those variables including the highly correlated variables.

Feature Engineering:

"MyProd1_Rx" "MyProd2_Rx" are prescription of our products.

“CompProd1_Rx” “CompProd2_Rx” are prescription of competitor products.

They are 4 variables which describes two characteristics.

We can add two variables which makes much sense.

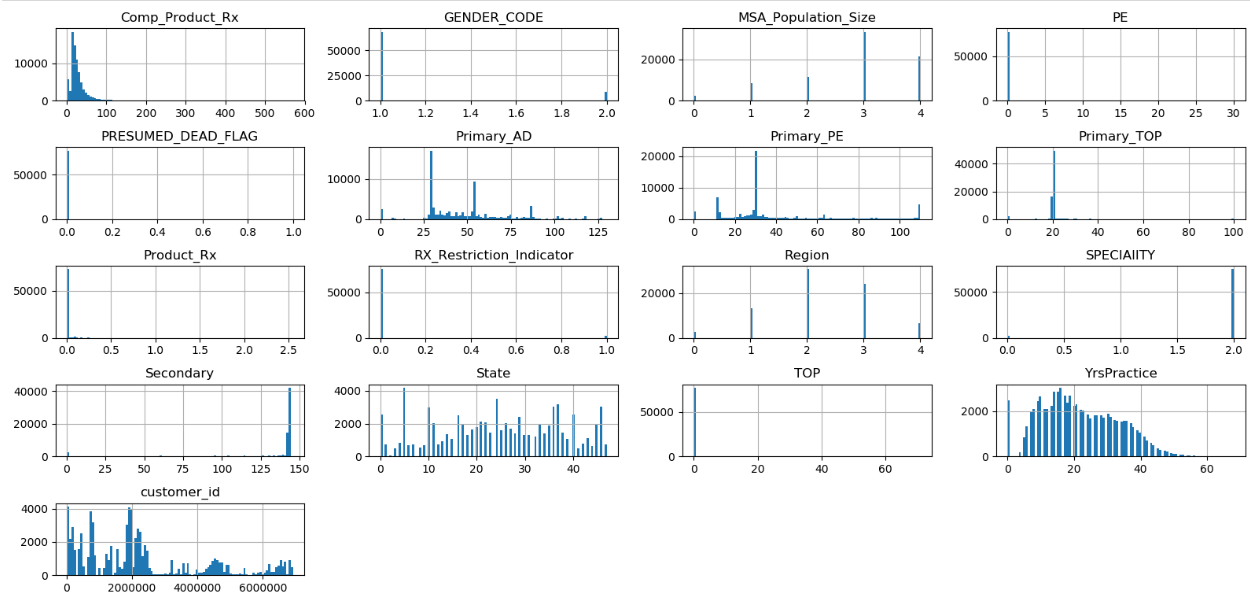
cleaned_data["Product_Rx"]=data["MyProd1_Rx"]+data["MyProd2_Rx"]

cleaned_data["Comp_Product_Rx"]=data["CompProd1_Rx"]+data["CompProd2_Rx"]

We have created two new variables to describe the characteristics of own products and competitor products.

Feature Scaling:

Let us check the distribution of all variables.



We can see that no variables has normal distribution. Hence we will use Normalization method to scale the variables.

Modelling:

K-Means Model:

We have applied K-Means model for the scaled data.

Let us check the results.

```
> summary(kmeans_model)
             Length Class  Mode
cluster      77247  -none- numeric
centers         64  -none- numeric
totss            1  -none- numeric
withinss         4  -none- numeric
tot.withinss     1  -none- numeric
betweenss        1  -none- numeric
size             4  -none- numeric
iter             1  -none- numeric
ifault           1  -none- numeric
> 
```

Row Labels	Sum of product_F	Sum of Comp_Product_F	Rate
High	126.5994539	257970.9962	0.000291
Low	400.4693923	618579.3155	0.000647
Medium	65.02638462	210509.605	0.000309
Super High	155.5255385	987317.7083	0.000158
Grand Total	**747.6207693**	**2074377.625**	

Rate=sum of product_rx/Sum of product_rx

Super High=0.000158

High=0.000291

Medium=0.000309

Low=0.000647

We can see that clusters are divided base on high sum_of_product_rx and less sum of comp_product