BHANU PRATAP REDDY

PREDICTIVE MODELLING
GRADED PROJECT

NOVEMBER 13 **2022**

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LINEAR REGRESSION PROBLEM

PROBLEM STATEMENT

The comp-activ databases is a collection of a computer systems activity measures.
The data was collected from a Sun Sparcstation 20/712 with 128 Mbytes of memory running in a multi-user university department. Users would typically be doing a large variety of tasks ranging from accessing the internet, editing files or running very CPU-bound programs.

As you are a budding data scientist you thought to find out a linear equation to build a model to predict 'usr'(Portion of time (%) that CPUs run in user mode) and to find out how each attribute affects the system to be in 'usr' mode using a list of system attributes.

DATA DICTIONARY

The data dictionary gives us information regarding the variables present tin the given dataset we are analysing and a brief explanation and what that variable contains or means. There fire the System measures used are given as:

- **lread** - Reads (transfers per second) between system memory and user memory
- **lwrite** - writes (transfers per second) between system memory and user memory
- **scall** - Number of system calls of all types per second
- **sread** - Number of system read calls per second .
- **swrite** - Number of system write calls per second .
- **fork** - Number of system fork calls per second.
- **exec** - Number of system exec calls per second.
- **rchar** - Number of characters transferred per second by system read calls
- **wchar** - Number of characters transfreed per second by system write calls
- **pgout** - Number of page out requests per second
- **ppgout** - Number of pages, paged out per second
- **pgfree** - Number of pages per second placed on the free list.
- **pgscan** - Number of pages checked if they can be freed per second

- **atch** - Number of page attaches (satisfying a page fault by reclaiming a page in memory) per second
- **pgin** - Number of page-in requests per second
- **ppgin** - Number of pages paged in per second
- **pflt** - Number of page faults caused by protection errors (copy-on-writes).
- **vflt** - Number of page faults caused by address translation.
- **runqsz** - Process run queue size (The number of kernel threads in memory that are waiting for a CPU to run. Typically, this value should be less than 2. Consistently higher values mean that the system might be CPU-bound.)
- **freemem** - Number of memory pages available to user processes
- **freeswap** - Number of disk blocks available for page swapping.
- **usr** - Portion of time (%) that cpus run in user mode

DATA INTIALIZATION AND PREPROCESSING

Data was initialized from given file and to verify that the data has been properly imported we look at the five head values of the data i.e., the first five rows.

	lread	lwrite	scall	sread	swrite	fork	exec	rchar	wchar	pgout	...	pgscan	atch	pgin	ppgin	pflt	vflt	runqsz	freemem	freeswap	usr
0	1	0	2147	79	68	0.2	0.2	40671.0	53995.0	0.0	...	0.0	0.0	1.6	2.6	16.00	26.40	CPU_Bound	4670	1730946	95
1	0	0	170	18	21	0.2	0.2	448.0	8385.0	0.0	...	0.0	0.0	0.0	0.0	15.63	16.83	Not_CPU_Bound	7278	1869002	97
2	15	3	2162	159	119	2.0	2.4	NaN	31950.0	0.0	...	0.0	1.2	6.0	9.4	150.20	220.20	Not_CPU_Bound	702	1021237	87
3	0	0	160	12	16	0.2	0.2	NaN	8670.0	0.0	...	0.0	0.0	0.2	0.2	15.60	16.80	Not_CPU_Bound	7248	1863704	98
4	5	1	330	39	38	0.4	0.4	NaN	12185.0	0.0	...	0.0	0.0	1.0	1.2	37.80	47.60	Not_CPU_Bound	633	1760253	90

Figure 1
First five data samples

It appears as if everything is loaded properly onto a dataframe. Now checking the overall size of the data, we get:

```
No. of Rows: 8192
No. of Columns: 22
```

Figure 2
Size of dataframe

It appears that the data has 8192 rows and 22 columns. Checking the data types of these columns:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8192 entries, 0 to 8191
Data columns (total 22 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   lread     8192 non-null   int64
 1   lwrite    8192 non-null   int64
 2   scall     8192 non-null   int64
 3   sread     8192 non-null   int64
 4   swrite    8192 non-null   int64
 5   fork      8192 non-null   float64
 6   exec      8192 non-null   float64
 7   rchar     8088 non-null   float64
 8   wchar     8177 non-null   float64
 9   pgout     8192 non-null   float64
 10  ppgout    8192 non-null   float64
 11  pgfree    8192 non-null   float64
 12  pgscan    8192 non-null   float64
 13  atch      8192 non-null   float64
 14  pgin      8192 non-null   float64
 15  ppgin     8192 non-null   float64
 16  pflt      8192 non-null   float64
 17  vflt      8192 non-null   float64
 18  runqsz    8192 non-null   object
 19  freemem   8192 non-null   int64
 20  freeswap  8192 non-null   int64
 21  usr       8192 non-null   int64
dtypes: float64(13), int64(8), object(1)
```

Figure 3
Data type summary

Looking at the datatype info we can clearly see that there are a few missing values in certain columns and let us have a look at this in more detail:

```
lread         0
lwrite        0
scall         0
sread         0
swrite        0
fork          0
exec          0
rchar       104
wchar        15
pgout         0
ppgout        0
pgfree        0
pgscan        0
atch          0
pgin          0
ppgin         0
pflt          0
vflt          0
runqsz        0
freemem       0
freeswap      0
usr           0
```

Figure 4

Missing Values in dataframe.

It appears the columns 'rchar' and 'wchar' have missing values. Before we perform any kind of data preprocessing let us first have a look at the 5-point summary:

	count	mean	std	min	25%	50%	75%	max
lread	8192.0	1.955969e+01	53.353799	0.0	2.0	7.0	20.000	1845.00
lwrite	8192.0	1.310620e+01	29.891726	0.0	0.0	1.0	10.000	575.00
scall	8192.0	2.306318e+03	1633.617322	109.0	1012.0	2051.5	3317.250	12493.00
sread	8192.0	2.104800e+02	198.980146	6.0	86.0	166.0	279.000	5318.00
swrite	8192.0	1.500582e+02	160.478980	7.0	63.0	117.0	185.000	5456.00
fork	8192.0	1.884554e+00	2.479493	0.0	0.4	0.8	2.200	20.12
exec	8192.0	2.791998e+00	5.212456	0.0	0.2	1.2	2.800	59.56
rchar	8088.0	1.973857e+05	239837.493526	278.0	34091.5	125473.5	267828.750	2526649.00
wchar	8177.0	9.590299e+04	140841.707911	1498.0	22916.0	46619.0	106101.000	1801623.00
pgout	8192.0	2.285317e+00	5.307038	0.0	0.0	0.0	2.400	81.44
ppgout	8192.0	5.977229e+00	15.214590	0.0	0.0	0.0	4.200	184.20
pgfree	8192.0	1.191971e+01	32.363520	0.0	0.0	0.0	5.000	523.00
pgscan	8192.0	2.152685e+01	71.141340	0.0	0.0	0.0	0.000	1237.00
atch	8192.0	1.127505e+00	5.708347	0.0	0.0	0.0	0.600	211.58
pgin	8192.0	8.277960e+00	13.874978	0.0	0.6	2.8	9.765	141.20
ppgin	8192.0	1.238859e+01	22.281318	0.0	0.6	3.8	13.800	292.61
pflt	8192.0	1.097938e+02	114.419221	0.0	25.0	63.8	159.600	899.80
vflt	8192.0	1.853158e+02	191.000603	0.2	45.4	120.4	251.800	1365.00
freemem	8192.0	1.763456e+03	2482.104511	55.0	231.0	579.0	2002.250	12027.00
freeswap	8192.0	1.328126e+06	422019.426957	2.0	1042623.5	1289289.5	1730379.500	2243187.00
usr	8192.0	8.396887e+01	18.401905	0.0	81.0	89.0	94.000	99.00

Figure 5

5 point summary of dataframe

Nothing out of the ordinary in the 5-point summary, which we know will change after a bit of preprocessing.

Let us check for zero values in columns:

lread	675
lwrite	2684
scall	0
sread	0
swrite	0
fork	21
exec	21
rchar	0
wchar	0
pgout	4878
ppgout	4878
pgfree	4869
pgscan	6448
atch	4575
pgin	1220
ppgin	1220
pflt	3
vflt	0
runqsz	0
freemem	0
freeswap	0
usr	283

Figure 6
Number of 0 values in dataframe

Although there are quite a few 0 values in many columns the value being zero is of not a concern given the columns as there are situations and scenarios in which an event could lead to 0 value in the column. Hence, we will not be performing any preprocessing related to zero values in columns.

First we impute the null values in the data as seen in Fig. 4 with the respective median values of the columns before we further proceed (since number of missing values is less the 5% of data there is no concern of skew).
Hence null values measured after imputing:

lread	0
lwrite	0
scall	0
sread	0
swrite	0
fork	0
exec	0
rchar	0
wchar	0
pgout	0
ppgout	0
pgfree	0
pgscan	0
atch	0
pgin	0
ppgin	0
pflt	0
vflt	0
runqsz	0
freemem	0
freeswap	0
usr	0

Figure 7
Checking for null Values after treatment

Now let us Check for outliers in the data:

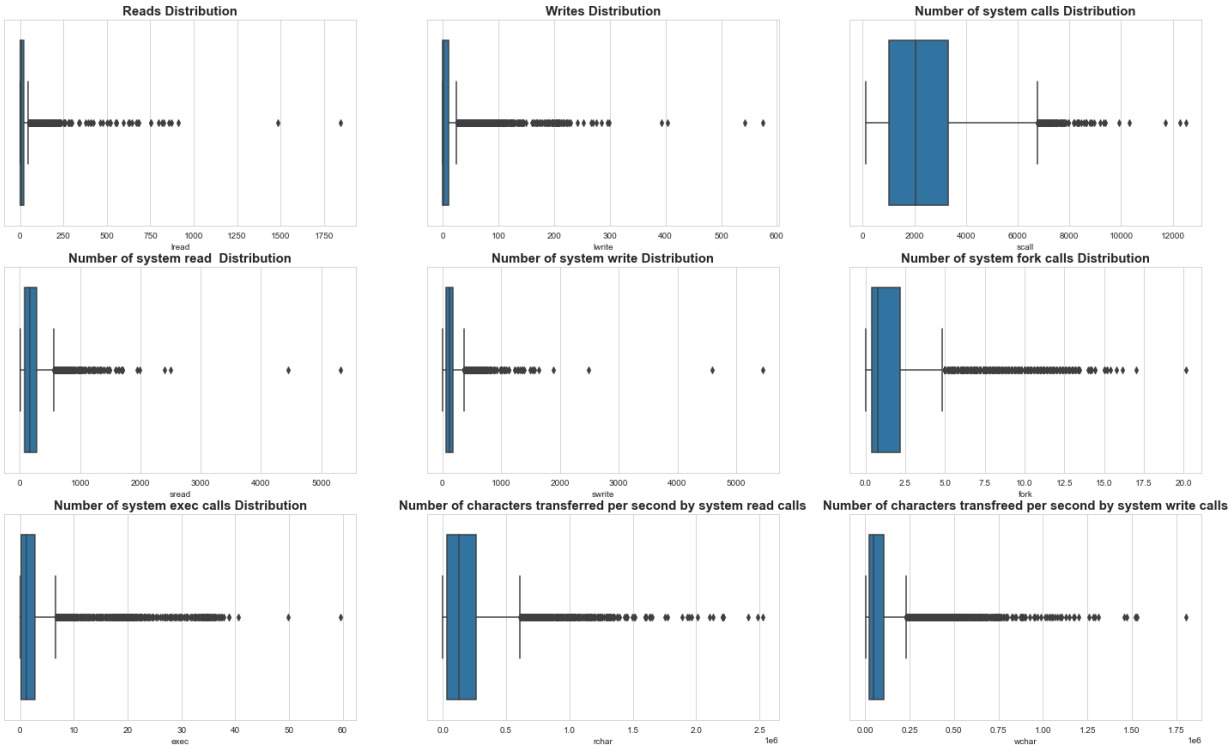

Figure 8 *Outliers Before Treatment - I*

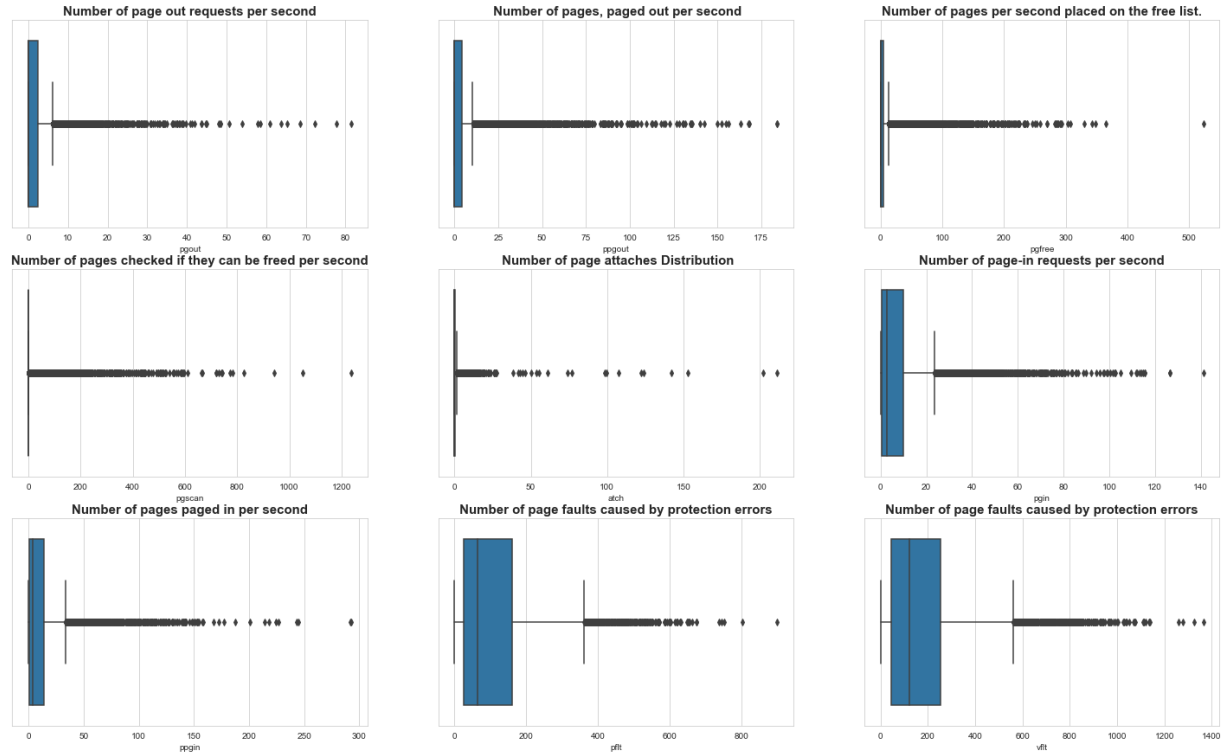

Figure 9 *Outliers Before Treatment - II*

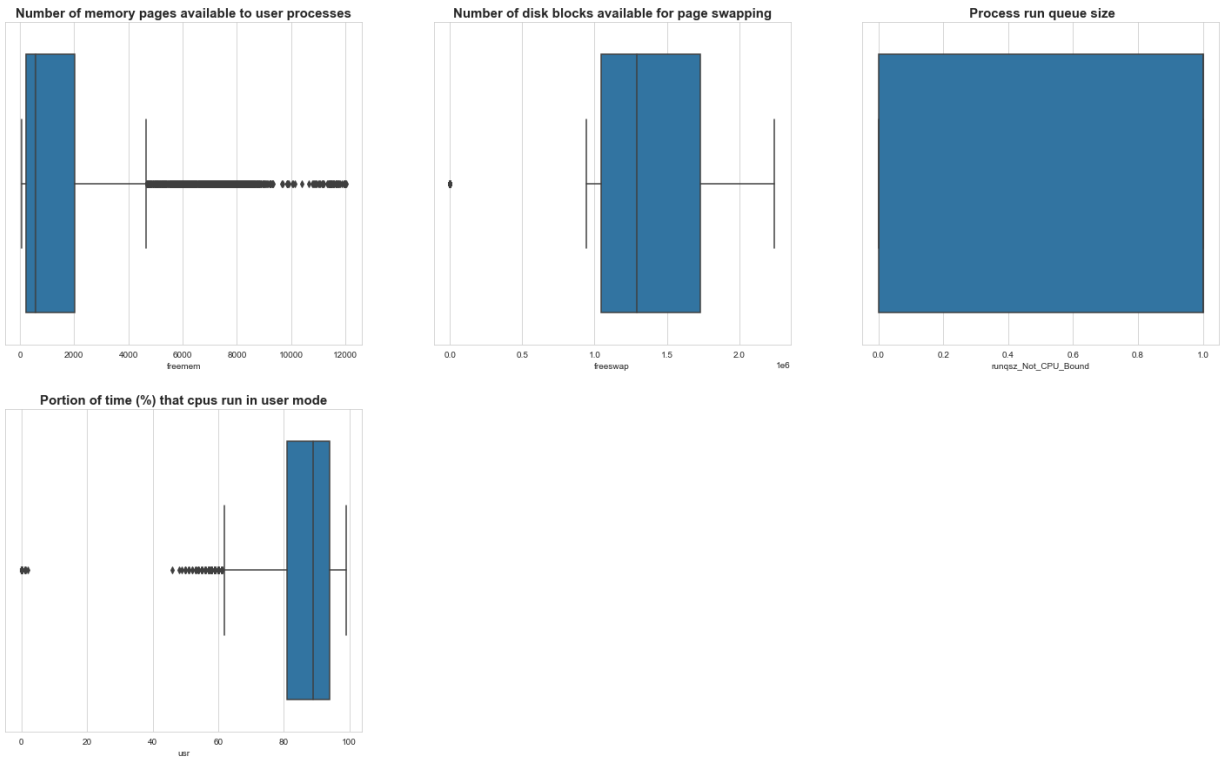

Figure 10 *Outliers Before Treatment - III*

It appears almost every column has outliers. Since, in linear regression Outliers can have a large effect on the output we have no choice but to treat them.

we will proceed with removing outliers by moving the outliers to its closest quartile.
The result after removing outliers:

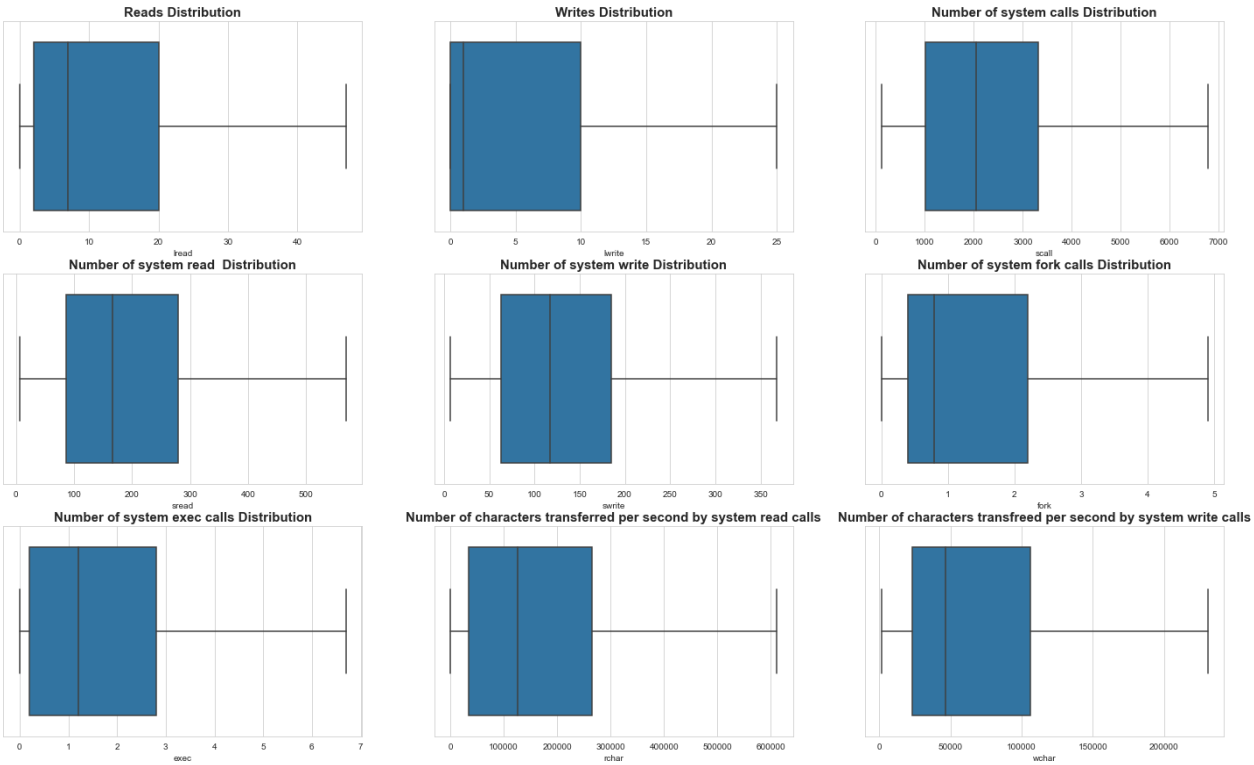

Figure 11 *Outliers After Treatment - I*

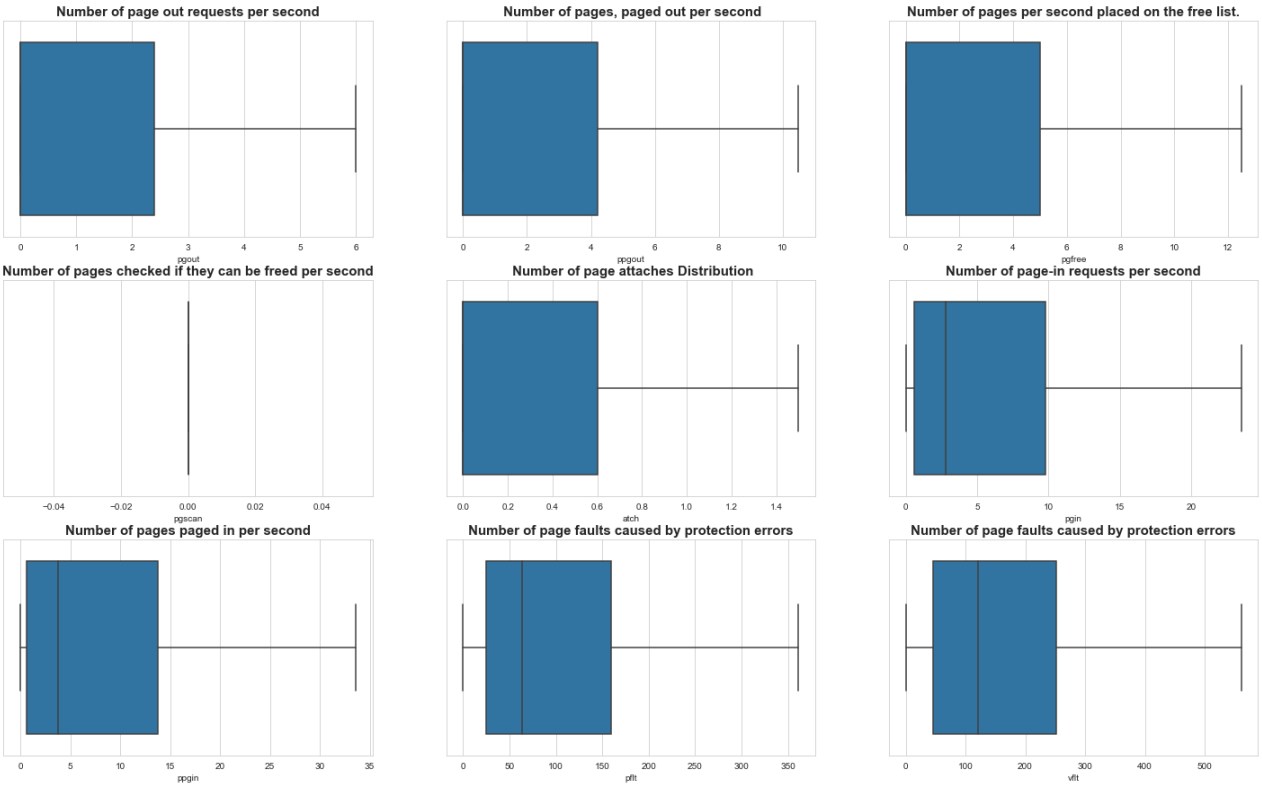

Figure 12 *Outliers After Treatment - II*

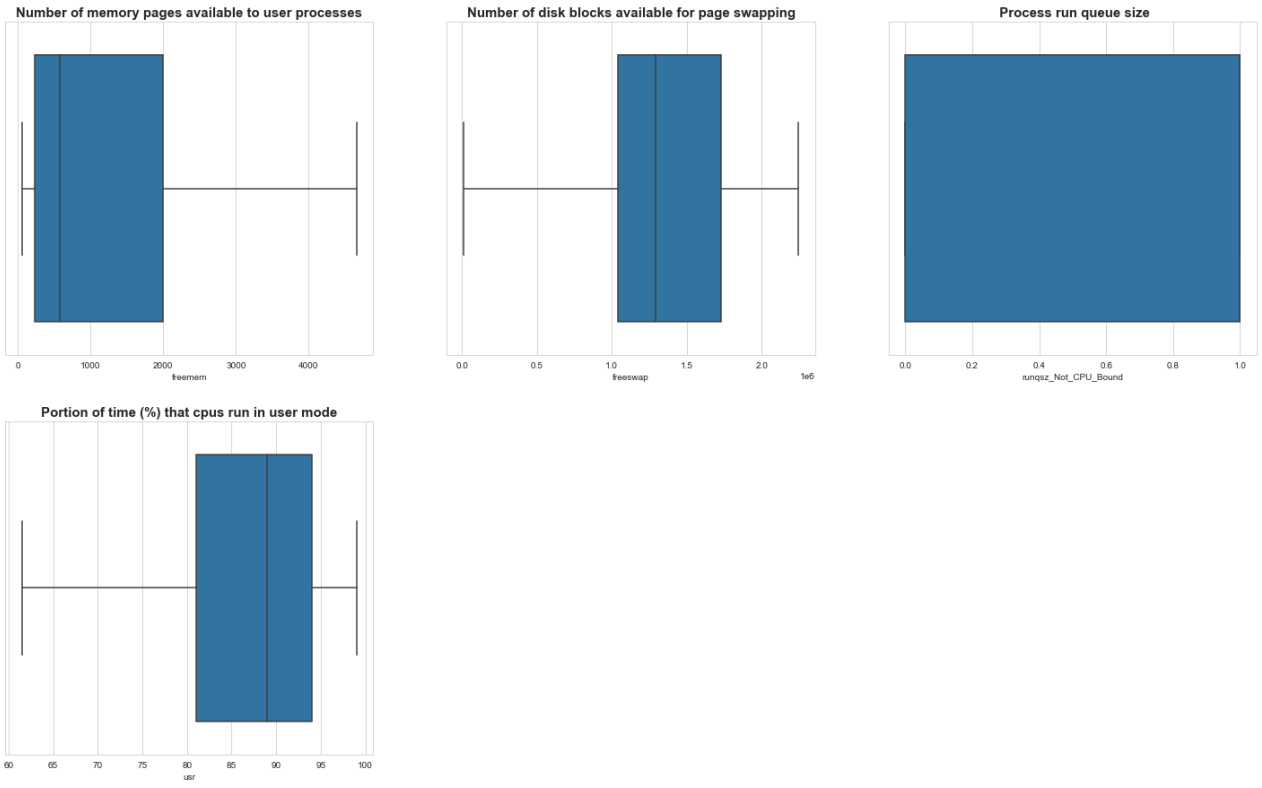

Figure 13 *Outliers After Treatment - III*

Let us look again at the 5-point summary after outlier treatment to see if there are any peculiarities.

	count	mean	std	min	25%	50%	75%	max
lread	8192.0	1.342285e+01	15.159741	0.0	2.00	7.0	20.000	4.700000e+01
lwrite	8192.0	6.657471e+00	9.291945	0.0	0.00	1.0	10.000	2.500000e+01
scall	8192.0	2.294484e+03	1593.093446	109.0	1012.00	2051.5	3317.250	6.775125e+03
sread	8192.0	1.997764e+02	146.758932	6.0	86.00	166.0	279.000	5.685000e+02
swrite	8192.0	1.379700e+02	97.141835	7.0	63.00	117.0	185.000	3.680000e+02
fork	8192.0	1.557771e+00	1.591220	0.0	0.40	0.8	2.200	4.900000e+00
exec	8192.0	1.931495e+00	2.028253	0.0	0.20	1.2	2.800	6.700000e+00
rchar	8192.0	1.788841e+05	174589.212910	278.0	34860.50	125473.5	265394.750	6.111961e+05
wchar	8192.0	7.564554e+04	71262.958027	1498.0	22977.75	46619.0	106037.000	2.306259e+05
pgout	8192.0	1.420901e+00	2.200251	0.0	0.00	0.0	2.400	6.000000e+00
ppgout	8192.0	2.560702e+00	4.037317	0.0	0.00	0.0	4.200	1.050000e+01
pgfree	8192.0	3.164586e+00	4.983345	0.0	0.00	0.0	5.000	1.250000e+01
pgscan	8192.0	0.000000e+00	0.000000	0.0	0.00	0.0	0.000	0.000000e+00
atch	8192.0	3.882788e-01	0.562937	0.0	0.00	0.0	0.600	1.500000e+00
pgin	8192.0	6.385262e+00	7.684420	0.0	0.60	2.8	9.765	2.351250e+01
ppgin	8192.0	9.140437e+00	11.160927	0.0	0.60	3.8	13.800	3.360000e+01
pflt	8192.0	1.056361e+02	101.548788	0.0	25.00	63.8	159.600	3.615000e+02
vflt	8192.0	1.756225e+02	162.497031	0.2	45.40	120.4	251.800	5.614000e+02
freemem	8192.0	1.387625e+03	1605.763418	55.0	231.00	579.0	2002.250	4.659125e+03
freeswap	8192.0	1.328520e+06	420782.723746	10989.5	1042623.50	1289289.5	1730379.500	2.243187e+06
usr	8192.0	8.624622e+01	9.748585	61.5	81.00	89.0	94.000	9.900000e+01
runqsz_Not_CPU_Bound	8192.0	5.286865e-01	0.499207	0.0	0.00	1.0	1.000	1.000000e+00

Figure 14
5 Point summary after outlier treatment

The one peculiar thing we see is that now 'pgscan' column has completely become 0. Since originally most of the pgscan column values were zero whatever data value other than 0 was recorded automatically becomes an outlier, but this is consistent with real world occurrences pgscan values is usually always zero.

However, we will not drop the column just for the sake of completeness for the linear regression algorithm (and since the data set is small it won't affect processing time) but the convention would be to drop pgscan before proceeding with linear regression as it does not affect the regression nor give any value.

The dataset was also checked for duplicated values and there were none.

```
False    8192
```

Figure 15
Duplicated data Indicator

Column 'runsqz' is of categorical in nature and since our ultimate goal is to perform linear regression categorical variables as is cannot be used. Hence, we convert subgroups to binary and drop one categorical value to remove redundancy.

EXPLORATORY DATA ANALYSIS (EDA)

Now that all the required preprocessing has completed, we can briefly explore the data to find any interesting trends or insights.

The most important aspect that we have to explore is the relationship between the numerical variables as that is what we would be most interested in with respect to linear regression. Hence the heatmap of the correlation among the variables can be given as:

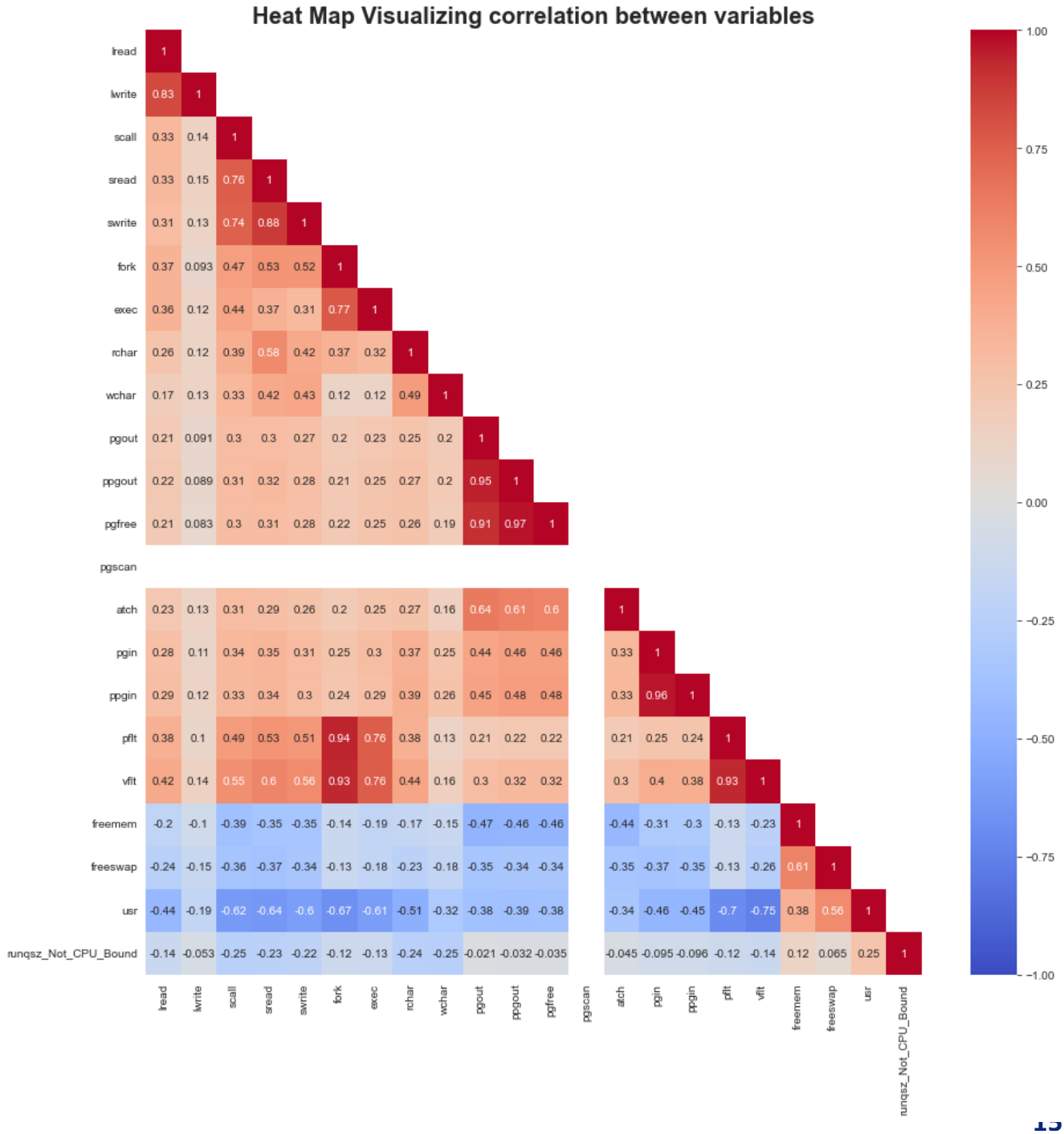

Figure 16

Heatmap depicting the correlation between all numerical variables

We can clearly see some correlation between many of the independent variables. But what is most interesting is the correlation of the dependent variable 'usr' to quite a few of the independent variables with the highest correlation to 'vlft'. We will further have a look at this after we finish building our model.

Similarly, we can try to visualize the bi variate relation between the variable with the help of a pair plot(Please refer to notebook for better clarity):

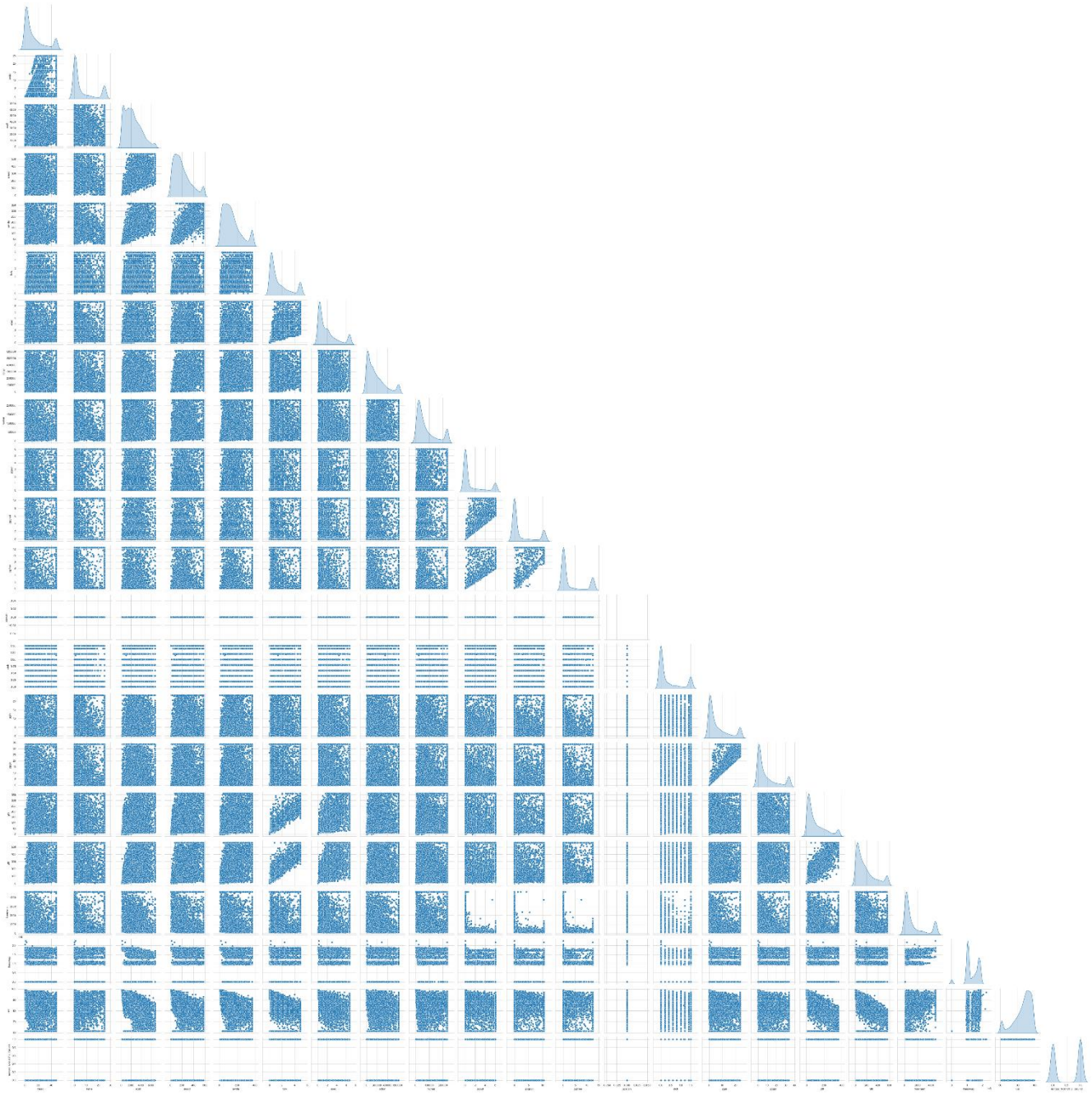

Figure 17

Pairplot depicting the bi variate plot between all numerical variables.

Let us have a closer look at the bivariate plots of vflt and pflt vs usr, given that they have the highest correlation:

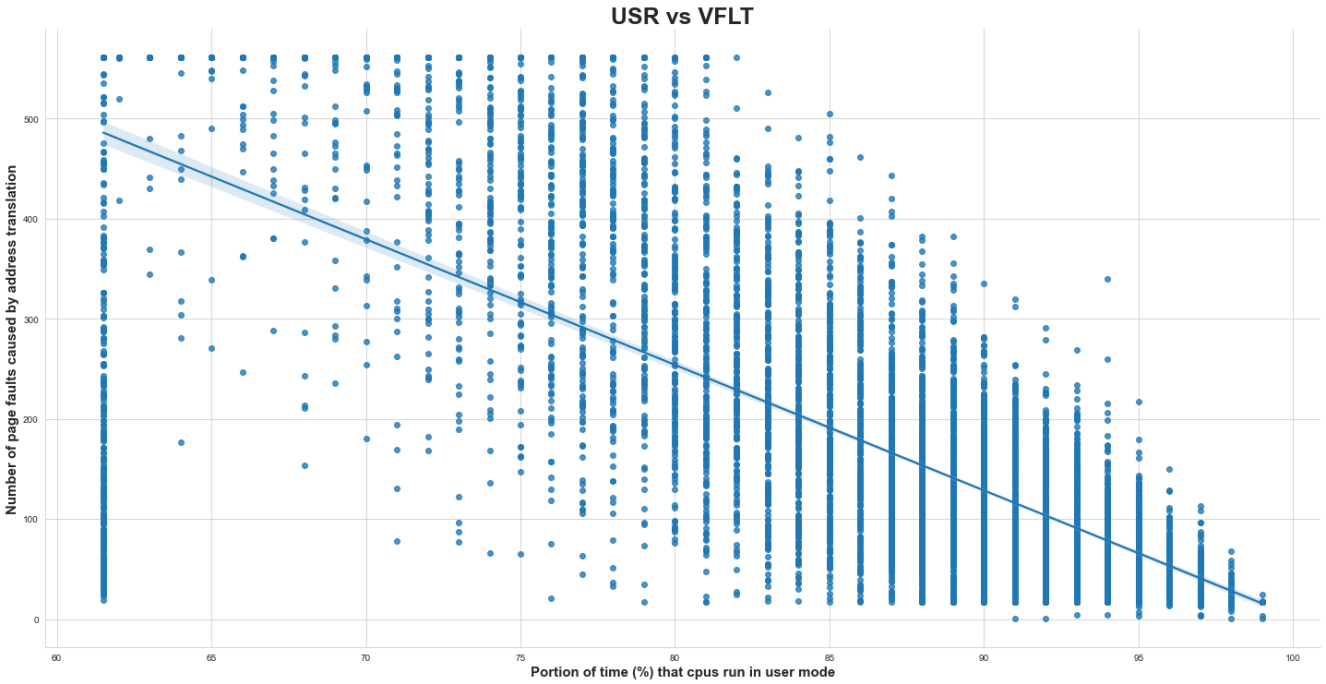

Figure 18
LMPlot of USR vs VFLT

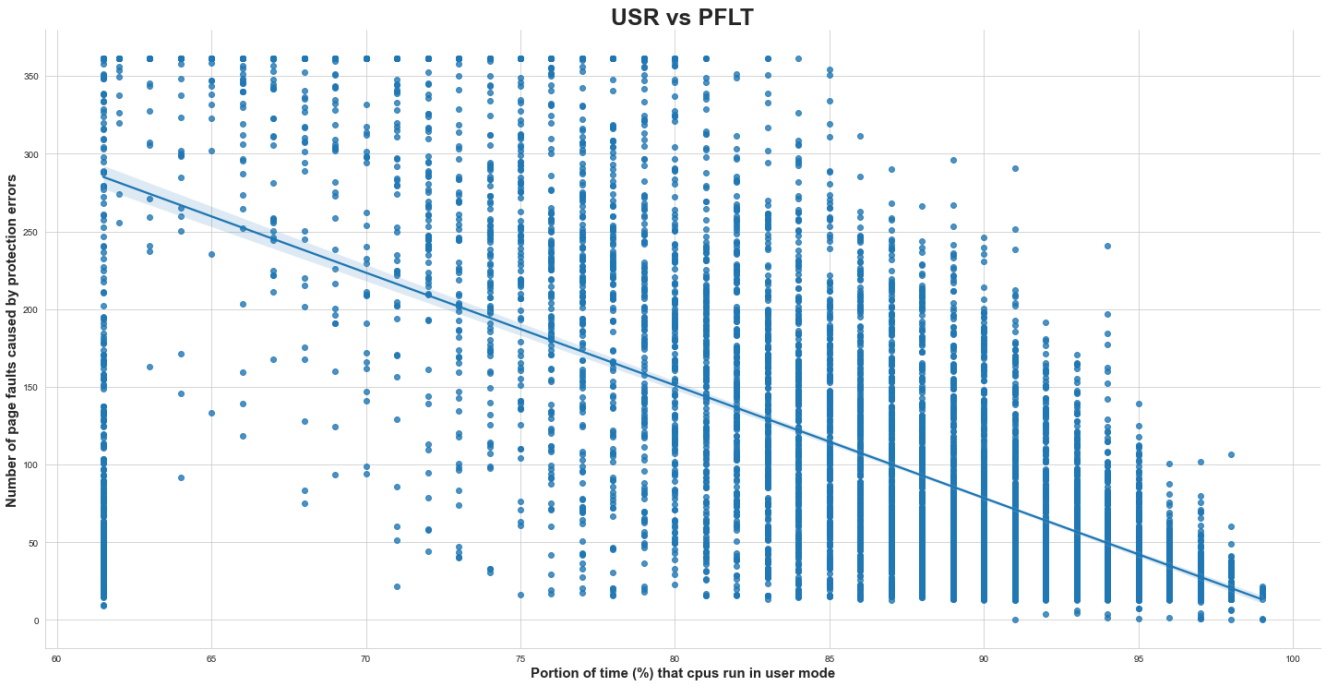

Figure 19
LMPlot of USR vs PFLT

They certainly do have a significant correlation, let us see how this is translated in linear regression.

Before we proceed with Linear regression let us have a quick look at the VIF to understand better the relationship of the variables and what would be the expectations of our process afterwards.

Table 1: *VIF Score before any Regression*

FEATURES	VIF SCORE
ppgout	43.019212
vflt	32.816397
usr	28.164636
fork	25.336356
freeswap	25.115743
pflt	24.296776
pgfree	24.111997
ppgin	23.35019
pgin	23.215483
sread	18.600202
swrite	16.967349
pgout	16.224461
lread	9.3273
scall	9.093957
lwrite	6.455932
exec	5.957179
rchar	4.268359
freemem	3.459706
wchar	3.402886
atch	2.750825
runqsz_Not_CPU_Bound	2.494291
pgscan	N/A

As we can see the VIF score is a bit high for a few variables. We will however not drop any of them now. We will have a closer look at this when we perform Linear Regression using stats model.

LINEAR REGRESSION - SCIKIT

First and foremost, we split the dependent variable and independent variables data into train and test in 70:30 ratio i.e.,

```
Number of rows and columns of the training set for the independent variables: (5734, 21)
Number of rows and columns of the training set for the dependent variable: (5734, 1)
Number of rows and columns of the test set for the independent variables: (2458, 21)
Number of rows and columns of the test set for the dependent variable: (2458, 1)
```

We apply the Linear Regression model on the train data and hence the coeffs we Obtain are as follows:

Table 2: *Table of Coeff of Scikit Linear Regression*

COLUMNS	COEFFICIENT ESTIMATE
Intercept	83.58331495
lread	-7.198099e-02
lwrite	6.303266e-02
scall	-6.894191e-04
sread	1.727363e-03
swrite	-5.471311e-03
fork	-1.064429e-01
exec	-2.830455e-01
rchar	-4.664703e-06
wchar	-6.306943e-06
pgout	-3.908755e-01
ppgout	-4.335392e-02
pgfree	9.343621e-02
pgscan	-5.551115e-16
atch	5.299278e-01
pgin	1.224288e-02
ppgin	-6.218414e-02
pflt	-3.154580e-02
vflt	-5.939833e-03
freemem	-4.472088e-04
freeswap	9.009683e-06
runqsz_Not_CPU_Bound	1.681395e+00

The above table can also be visualized as:

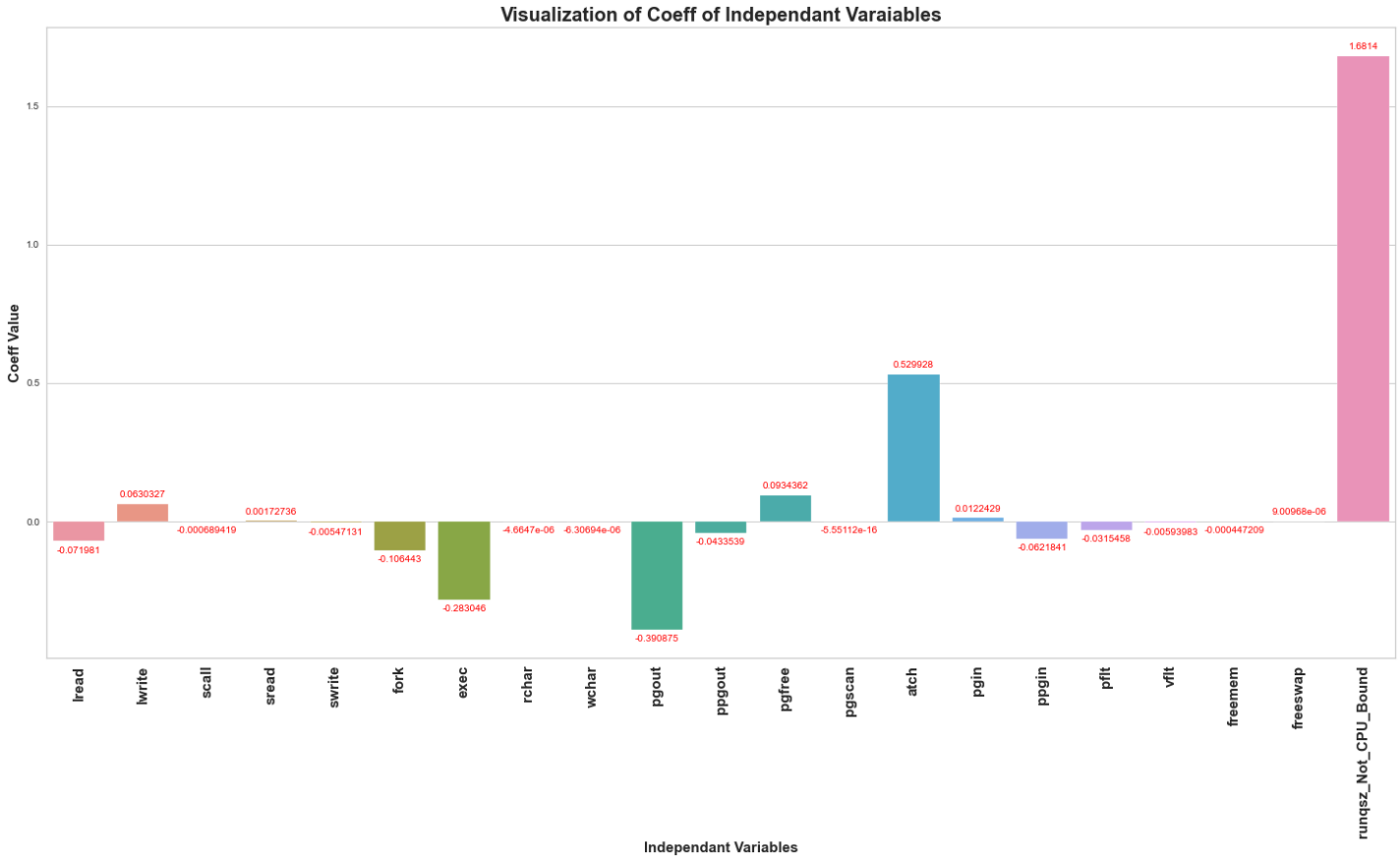

Figure 20

Visualization of Coeff of Independent variables

Before we comment on the intercepts let us have a look at the R^2 and RMSE values of train and test data:

Table 3 : *R^2 and RMSE of train and test data Scikit Linear regression*

	Train	Test
R^2	0.7881	0.7876
RMSE	4.4872	4.4536

Based upon the data seen above we can make the following statements:

- R^2 which is a measure of the fit of the regression model here amounts to 78.8% in train data and 78.6% in Test data which indicates that the model is a good fit without any overfitting or underfitting.

- The RMSE is the square root of the variance of the residuals. It indicates the absolute fit of the model to the data–how close the observed data points are to the model's predicted values. Whereas R-squared is a relative measure of fit, RMSE is an absolute measure of fit. As the square root of a variance, RMSE can be interpreted as the standard deviation of the unexplained variance. It has the useful property of being in the same units as the response variable. Lower values of RMSE indicate better fit. The RMSE Here appears a bit high indicating an overfit but we shall compare it with the stats model which will help us give a better understanding.

LINEAR REGRESSION – STATS MODEL

Utilizing the same split data as from the scikit model, we apply Liner regression through the stats model with all dependent variables included.
The result:

Table 4: *Table of Coeff of stats model Linear Regression*

COLUMNS	COEFFICIENT ESTIMATE
Intercept	8.358331E+01
lread	-7.198099E-02
lwrite	6.303266E-02
scall	-6.894191E-04
sread	1.727363E-03
swrite	-5.471311E-03
fork	-1.064429E-01
exec	-2.830455E-01
rchar	-4.664703E-06
wchar	-6.306943E-06
pgout	-3.908755E-01
ppgout	-4.335392E-02
pgfree	9.343621E-02
pgscan	-6.137913E-15
atch	5.299278E-01
pgin	1.224288E-02
ppgin	-6.218414E-02
pflt	-3.154580E-02
vflt	-5.939833E-03
freemem	-4.472088E-04
freeswap	9.009683E-06

This is very similar to the results we obtained through regression from scikit. No let us have a closer look at the OLS regression results summary report, based on which we will further process the data.

OLS Regression Results

Dep. Variable:	usr	R-squared:	0.788
Model:	OLS	Adj. R-squared:	0.787
Method:	Least Squares	F-statistic:	1063.
Date:	Sat, 12 Nov 2022	Prob (F-statistic):	0.00
Time:	15:37:52	Log-Likelihood:	-16744.
No. Observations:	5734	AIC:	3.353e+04
Df Residuals:	5713	BIC:	3.367e+04
Df Model:	20		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	83.5833	0.316	264.794	0.000	82.965	84.202
lread	-0.0720	0.009	-8.110	0.000	-0.089	-0.055
lwrite	0.0630	0.013	4.811	0.000	0.037	0.089
scall	-0.0007	6.52e-05	-10.571	0.000	-0.001	-0.001
sread	0.0017	0.001	1.668	0.095	-0.000	0.004
swrite	-0.0055	0.001	-3.776	0.000	-0.008	-0.003
fork	-0.1064	0.134	-0.792	0.428	-0.370	0.157
exec	-0.2830	0.052	-5.401	0.000	-0.386	-0.180
rchar	-4.665e-06	4.92e-07	-9.488	0.000	-5.63e-06	-3.7e-06
wchar	-6.307e-06	1.06e-06	-5.973	0.000	-8.38e-06	-4.24e-06
pgout	-0.3909	0.094	-4.142	0.000	-0.576	-0.206
ppgout	-0.0434	0.086	-0.506	0.613	-0.211	0.125
pgfree	0.0934	0.052	1.800	0.072	-0.008	0.195
pgscan	-6.138e-15	8.34e-17	-73.591	0.000	-6.3e-15	-5.97e-15
atch	0.5299	0.145	3.661	0.000	0.246	0.814
pgin	0.0122	0.029	0.426	0.670	-0.044	0.069
ppgin	-0.0622	0.020	-3.117	0.002	-0.101	-0.023
pflt	-0.0315	0.002	-15.908	0.000	-0.035	-0.028
vflt	-0.0059	0.001	-4.124	0.000	-0.009	-0.003
freemem	-0.0004	5.17e-05	-8.645	0.000	-0.001	-0.000
freeswap	9.01e-06	1.91e-07	47.140	0.000	8.64e-06	9.38e-06
runqsz_Not_CPU_Bound	1.6814	0.127	13.215	0.000	1.432	1.931

Omnibus:	1098.987	Durbin-Watson:	1.962
Prob(Omnibus):	0.000	Jarque-Bera (JB):	2350.161
Skew:	-1.117	Prob(JB):	0.00
Kurtosis:	5.202	Cond. No.	2.47e+23

Figure 21

Regression Results stats Model - I

The first thing we must observe is the probability value of $t_{critical}$ of all the coefficients. As the summary indicates there seems to be a problem of multicollinearity which we can address by removing the independent variables with a probability higher than critical value i.e., 0.05. We must also do this one at a time and generate the OLS after each removal until all probabilities are lower than critical value.

We delete ‘pgin’ and run the model again:

OLS Regression Results

Dep. Variable:	usr	R-squared:	0.788
Model:	OLS	Adj. R-squared:	0.787
Method:	Least Squares	F-statistic:	1119.
Date:	Sat, 12 Nov 2022	Prob (F-statistic):	0.00
Time:	15:37:53	Log-Likelihood:	-16744.
No. Observations:	5734	AIC:	3.353e+04
Df Residuals:	5714	BIC:	3.366e+04
Df Model:	19		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	83.5932	0.315	265.559	0.000	82.976	84.210
lread	-0.0721	0.009	-8.131	0.000	-0.090	-0.055
lwrite	0.0631	0.013	4.817	0.000	0.037	0.089
scall	-0.0007	6.52e-05	-10.563	0.000	-0.001	-0.001
sread	0.0017	0.001	1.664	0.096	-0.000	0.004
swrite	-0.0055	0.001	-3.779	0.000	-0.008	-0.003
fork	-0.1079	0.134	-0.803	0.422	-0.371	0.155
exec	-0.2825	0.052	-5.393	0.000	-0.385	-0.180
rchar	-4.678e-06	4.91e-07	-9.536	0.000	-5.64e-06	-3.72e-06
wchar	-6.3e-06	1.06e-06	-5.967	0.000	-8.37e-06	-4.23e-06
pgout	-0.3888	0.094	-4.126	0.000	-0.574	-0.204
ppgout	-0.0443	0.086	-0.518	0.605	-0.212	0.123
pgfree	0.0930	0.052	1.793	0.073	-0.009	0.195
pgscan	1.16e-14	6.62e-17	175.212	0.000	1.15e-14	1.17e-14
atch	0.5302	0.145	3.663	0.000	0.246	0.814
ppgin	-0.0542	0.007	-7.720	0.000	-0.068	-0.040
pflt	-0.0316	0.002	-15.959	0.000	-0.035	-0.028
vflt	-0.0059	0.001	-4.102	0.000	-0.009	-0.003
freemem	-0.0004	5.17e-05	-8.650	0.000	-0.001	-0.000
freeswap	9.005e-06	1.91e-07	47.208	0.000	8.63e-06	9.38e-06
runqsz_Not_CPU_Bound	1.6819	0.127	13.221	0.000	1.433	1.931

Omnibus:	1099.965	Durbin-Watson:	1.962
Prob(Omnibus):	0.000	Jarque-Bera (JB):	2353.904
Skew:	-1.117	Prob(JB):	0.00
Kurtosis:	5.205	Cond. No.	4.91e+22

Figure 22

Regression Results stats Model - II

We repeat the process for ‘ppgout’.

We delete ‘ppgout’ and run the model again:

OLS Regression Results

Dep. Variable:	usr	R-squared:	0.788
Model:	OLS	Adj. R-squared:	0.787
Method:	Least Squares	F-statistic:	1181.
Date:	Sat, 12 Nov 2022	Prob (F-statistic):	0.00
Time:	15:37:53	Log-Likelihood:	-16745.
No. Observations:	5734	AIC:	3.353e+04
Df Residuals:	5715	BIC:	3.365e+04
Df Model:	18		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	83.6060	0.314	266.447	0.000	82.991	84.221
lread	-0.0722	0.009	-8.149	0.000	-0.090	-0.055
lwrite	0.0632	0.013	4.828	0.000	0.038	0.089
scall	-0.0007	6.52e-05	-10.563	0.000	-0.001	-0.001
sread	0.0017	0.001	1.660	0.097	-0.000	0.004
swrite	-0.0055	0.001	-3.781	0.000	-0.008	-0.003
fork	-0.1064	0.134	-0.792	0.428	-0.370	0.157
exec	-0.2829	0.052	-5.401	0.000	-0.386	-0.180
rchar	-4.682e-06	4.9e-07	-9.547	0.000	-5.64e-06	-3.72e-06
wchar	-6.327e-06	1.05e-06	-6.000	0.000	-8.39e-06	-4.26e-06
pgout	-0.4215	0.070	-6.033	0.000	-0.558	-0.285
pgfree	0.0713	0.030	2.340	0.019	0.012	0.131
pgscan	3.768e-14	1.56e-16	241.611	0.000	3.74e-14	3.8e-14
atch	0.5332	0.145	3.687	0.000	0.250	0.817
ppgin	-0.0544	0.007	-7.752	0.000	-0.068	-0.041
pflt	-0.0316	0.002	-15.965	0.000	-0.035	-0.028
vflt	-0.0059	0.001	-4.105	0.000	-0.009	-0.003
freemem	-0.0004	5.17e-05	-8.673	0.000	-0.001	-0.000
freeswap	9.001e-06	1.91e-07	47.220	0.000	8.63e-06	9.38e-06
runqsz_Not_CPU_Bound	1.6807	0.127	13.215	0.000	1.431	1.930

Omnibus:	1099.263	Durbin-Watson:	1.961
Prob(Omnibus):	0.000	Jarque-Bera (JB):	2351.390
Skew:	-1.117	Prob(JB):	0.00
Kurtosis:	5.203	Cond. No.	1.80e+22

Figure 23

Regression Results stats Model - III

We repeat the process for ‘fork.

We delete ‘fork’ and run the model again:

OLS Regression Results

Dep. Variable:	usr	R-squared:	0.788
Model:	OLS	Adj. R-squared:	0.787
Method:	Least Squares	F-statistic:	1250.
Date:	Sat, 12 Nov 2022	Prob (F-statistic):	0.00
Time:	15:37:53	Log-Likelihood:	-16745.
No. Observations:	5734	AIC:	3.353e+04
Df Residuals:	5716	BIC:	3.365e+04
Df Model:	17		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	83.6376	0.311	268.733	0.000	83.027	84.248
lread	-0.0725	0.009	-8.185	0.000	-0.090	-0.055
lwrite	0.0639	0.013	4.889	0.000	0.038	0.090
scall	-0.0007	6.47e-05	-10.547	0.000	-0.001	-0.001
sread	0.0018	0.001	1.720	0.086	-0.000	0.004
swrite	-0.0057	0.001	-4.027	0.000	-0.008	-0.003
exec	-0.2950	0.050	-5.887	0.000	-0.393	-0.197
rchar	-4.688e-06	4.9e-07	-9.559	0.000	-5.65e-06	-3.73e-06
wchar	-6.307e-06	1.05e-06	-5.983	0.000	-8.37e-06	-4.24e-06
pgout	-0.4218	0.070	-6.037	0.000	-0.559	-0.285
pgfree	0.0717	0.030	2.353	0.019	0.012	0.131
pgscan	3.032e-14	1.67e-16	181.464	0.000	3e-14	3.06e-14
atch	0.5418	0.144	3.757	0.000	0.259	0.824
ppgin	-0.0534	0.007	-7.734	0.000	-0.067	-0.040
pflt	-0.0322	0.002	-17.459	0.000	-0.036	-0.029
vflt	-0.0064	0.001	-5.151	0.000	-0.009	-0.004
freemem	-0.0004	5.17e-05	-8.671	0.000	-0.001	-0.000
freeswap	8.983e-06	1.89e-07	47.475	0.000	8.61e-06	9.35e-06
runqsz_Not_CPU_Bound	1.6793	0.127	13.205	0.000	1.430	1.929

Omnibus:	1100.935	Durbin-Watson:	1.962
Prob(Omnibus):	0.000	Jarque-Bera (JB):	2363.584
Skew:	-1.117	Prob(JB):	0.00
Kurtosis:	5.215	Cond. No.	7.10e+21

Figure 24

Regression Results stats Model - IV

We repeat the process for ‘sread’.

We delete ‘sread’ and run the model again:

OLS Regression Results

Dep. Variable:	usr	R-squared:	0.788
Model:	OLS	Adj. R-squared:	0.787
Method:	Least Squares	F-statistic:	1328.
Date:	Sat, 12 Nov 2022	Prob (F-statistic):	0.00
Time:	15:37:53	Log-Likelihood:	-16746.
No. Observations:	5734	AIC:	3.353e+04
Df Residuals:	5717	BIC:	3.364e+04
Df Model:	16		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	83.6629	0.311	269.070	0.000	83.053	84.272
lread	-0.0733	0.009	-8.287	0.000	-0.091	-0.056
lwrite	0.0652	0.013	4.999	0.000	0.040	0.091
scall	-0.0006	6.12e-05	-10.554	0.000	-0.001	-0.001
swrite	-0.0041	0.001	-3.837	0.000	-0.006	-0.002
exec	-0.3002	0.050	-6.000	0.000	-0.398	-0.202
rchar	-4.324e-06	4.43e-07	-9.771	0.000	-5.19e-06	-3.46e-06
wchar	-6.472e-06	1.05e-06	-6.165	0.000	-8.53e-06	-4.41e-06
pgout	-0.4243	0.070	-6.073	0.000	-0.561	-0.287
pgfree	0.0743	0.030	2.440	0.015	0.015	0.134
pgscan	8.765e-14	3.5e-16	250.163	0.000	8.7e-14	8.83e-14
atch	0.5327	0.144	3.696	0.000	0.250	0.815
ppgin	-0.0541	0.007	-7.840	0.000	-0.068	-0.041
pflt	-0.0322	0.002	-17.461	0.000	-0.036	-0.029
vflt	-0.0063	0.001	-5.046	0.000	-0.009	-0.004
freemem	-0.0004	5.17e-05	-8.647	0.000	-0.001	-0.000
freeswap	8.958e-06	1.89e-07	47.478	0.000	8.59e-06	9.33e-06
runqsz_Not_CPU_Bound	1.6788	0.127	13.199	0.000	1.429	1.928

Omnibus:	1105.082	Durbin-Watson:	1.961
Prob(Omnibus):	0.000	Jarque-Bera (JB):	2377.872
Skew:	-1.120	Prob(JB):	0.00
Kurtosis:	5.222	Cond. No.	1.46e+22

Figure 25

Regression Results stats Model - V

Now let us have a look at how the VIF looks for all the independent variables.

Table 5: *Final VIF table after processing.*

FEATURES	VIF SCORE
usr	27.392986
vflt	25.25232
freeswap	24.523281
pflt	21.143251
pgout	9.316051
swrite	9.303795
lread	8.903767
pgfree	8.891434
scall	8.125213
lwrite	6.101878
exec	5.522918
freemem	3.453586
rchar	3.389531
wchar	3.31116
ppgin	2.744222
atch	2.687555
runqsz_Not_CPU_Bound	2.479421
pgscan	N/A

From what we gather above it seems we have reduced the VIF score significantly. And the RMS Score was calculated to be **4.48**, We can further analyze the data to see if this is enough to proceed on.

Let us have a look at the Predicted data vs Residuals:

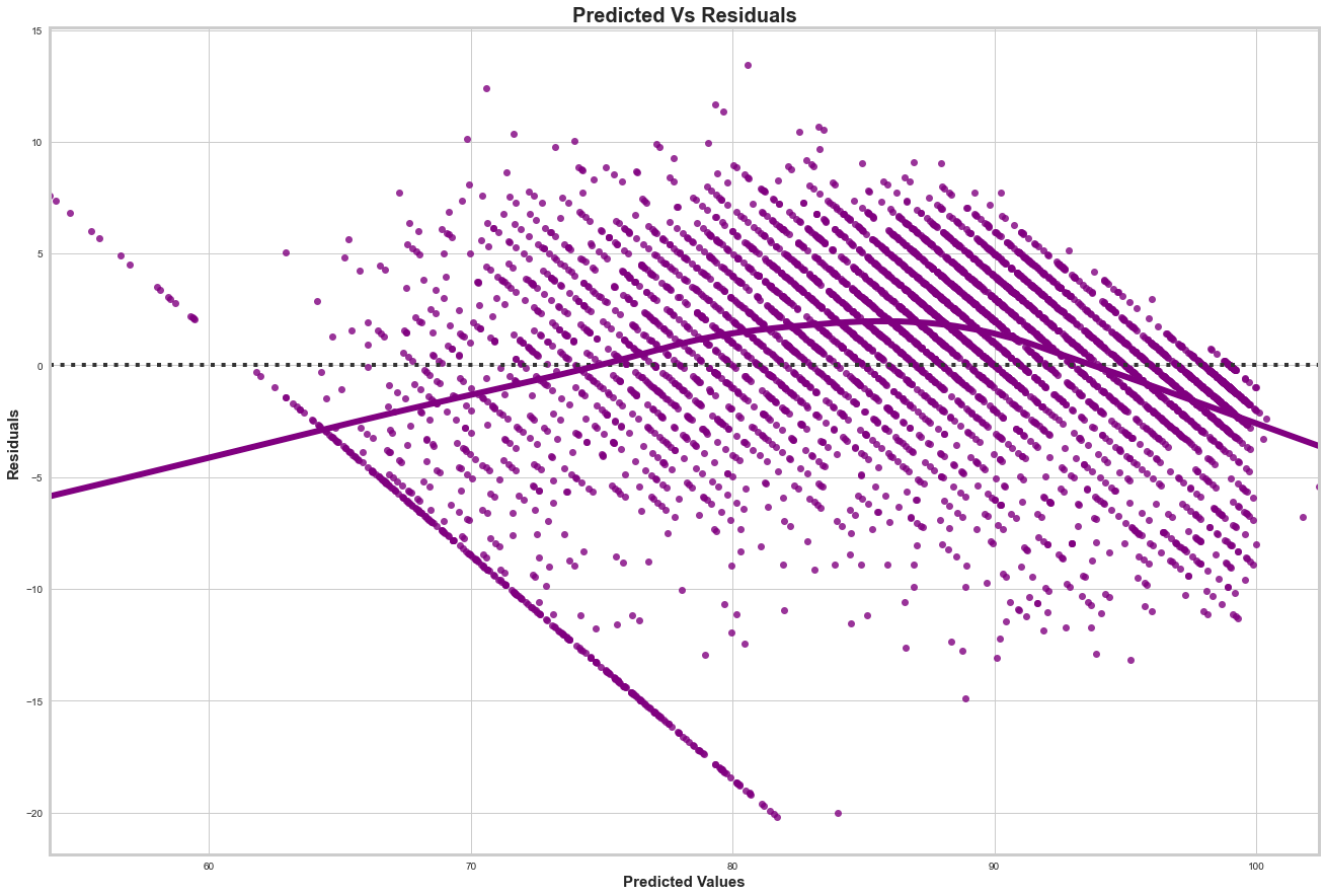

Figure 26

Predicted Values vs Residual Values Visualization

As it can be clearly seen there is an element of non-linearity in the distribution which indicates that given our data is obviously nonlinear the regression model obtained might not be the best representation. Hence, we must look to transform the independent variables as per their relationship with the target variable (But this does not fall withing the purview of this exercise).

Let us also check for normality within the residuals:

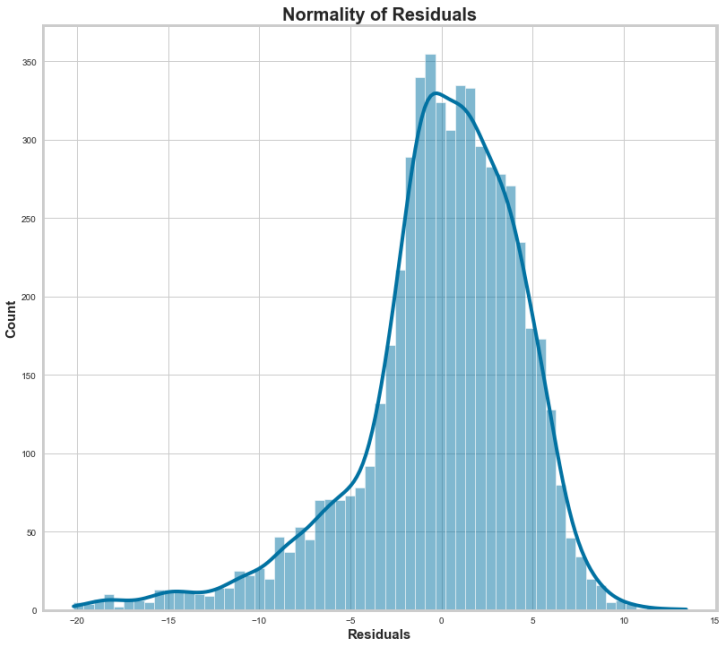

Figure 27
Normality of Residuals Distribution

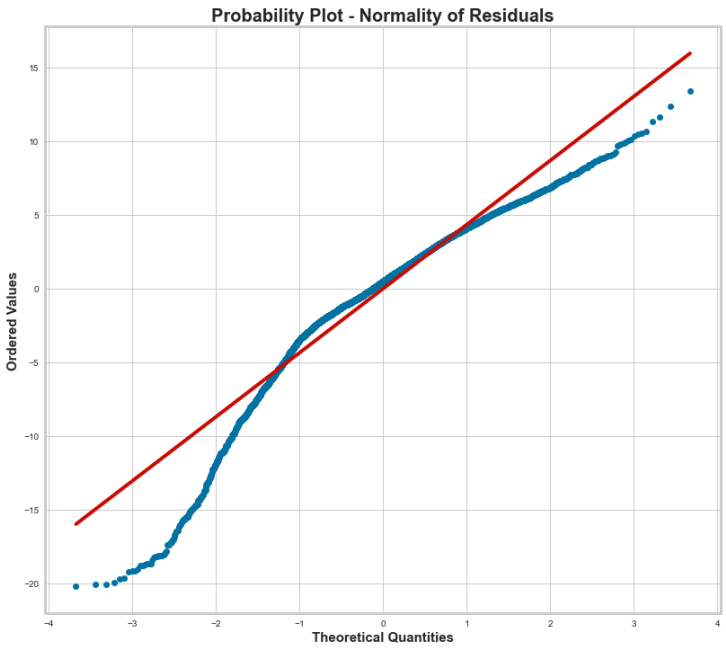

Figure 28
QQ Plot for residuals

The above plots clearly show that the data is not a perfect normal distribution, rather it has a certain skew to it. In order to address this problem, we can take transformations of the data i.e. scale it to log,exp…. To address this (This again does not fall withing the purview of this exercise).

Now let us have a look at the actual vs Predicted for the dependent variable.

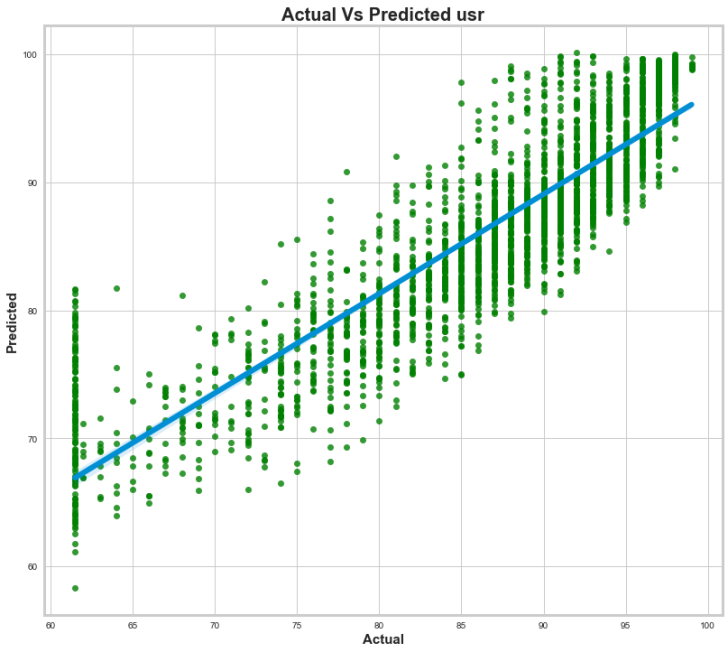

Figure 29
Actual vs Predicted for Dependent Variable

The Final Regression Equation Can be given as:

(83.66) * Intercept + (-0.07) * lread + (0.07) * lwrite + (-0.0) * scall + (-0.0) * swrite + (-0.3) * exec + (-0.0) * rchar + (-0.0) * wchar + (-0.42) * pgout + (0.07) * pgfree + (0.0) * pgscan + (0.53) * atch + (-0.05) * ppgin + (-0.03) * pflt + (-0.01) * vflt + (-0.0) * freemem + (0.0) * freeswap + (1.68) * runqsz_Not_CPU_Bound

Hence the Final Coeff Table:

Table 6: *Final Table of Coeff after processing.*

COLUMNS	COEFFICIENT ESTIMATE
Intercept	83.66294
lread	-0.07332106
lwrite	0.06523667
scall	-0.000646152
swrite	-0.004114926
exec	-0.3001651
rchar	-4.32423E-06
wchar	-6.47246E-06
pgout	-0.4242913
pgfree	0.0742597
pgscan	8.76473E-14
atch	0.5326521
ppgin	-0.05407726
pflt	-0.03218406
vflt	-0.006288155
freemem	-0.000447014
freeswap	8.95762E-06
runqsz_Not_CPU_Bound	1.678778

The above table can be visually represented as (Excluding Intercept):

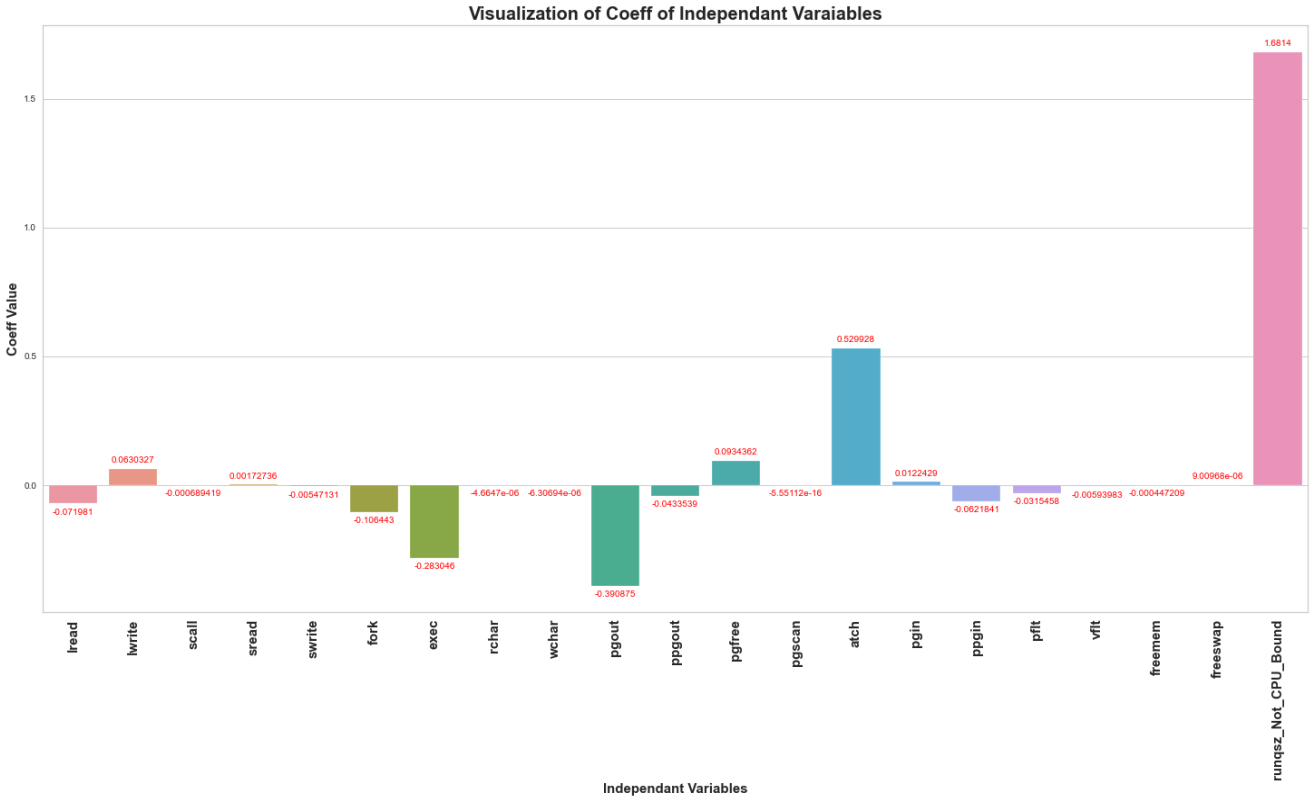

Figure 30

Visualization of final table of coeff

The key takeaways form this representation:

- ‘runqsz’, ‘atch’, ‘pgout’ and ‘exec’ are the parameters with the most influence on the model.
- Meaning change in any one of ‘runqsz’, ‘atch’, ‘pgout’ and ‘exec’ results in a higher magnitude of change that effects the output i.e., ‘usr’
- ‘runqsz’ and ‘atch’ have a positive correlation to change wrt ‘usr’.
- ‘pgout’ and ‘exec’ have a negative correlation to change wrt ‘usr’

SUMMARY, INSIGHTS & RECOMMENDATIONS

- From the process above we can state that The Process run que size and Number of page attaches when increased also increase the % of time run in user mode significantly.

- Therefore, in order to maintain a low user mode % either it must be made sure that the run que size is limited along with page attaches or else we increase the number of page out requests per second and number of system exec calls per second (i.e., increase processing capability)

- The given data is also has a lot of Multicollinearity meaning that whatever linear response we derive for this data it will have a high amount of inaccuracy since its response is most definitely quadratic (as evidenced in the residuals plots) in nature hence no amount of linear representation is good enough.

LOGISTIC, LDA & CART PROBLEM

PROBLEM STATEMENT

You are a statistician at the Republic of Indonesia Ministry of Health, and you are provided with a data of 1473 females collected from a Contraceptive Prevalence Survey. The samples are married women who were either not pregnant or do not know if they were at the time of the survey.

The problem is to predict do/don't they use a contraceptive method of choice based on their demographic and socio-economic characteristics.

DATA DICTIONARY

- **Wife_age** - Wife's age (numerical)
- **Wife_ education** - Wife's education (categorical) 1=uneducated, 2, 3, 4=tertiary
- **Husband_education** - Husband's education (categorical) 1=uneducated, 2, 3, 4=tertiary
- **No_of_children_born** - Number of children ever born (numerical)
- **Wife_religion** - Wife's religion (binary) Non-Scientology, Scientology
- **Wife_Working** - Wife's now working? (binary) Yes, No
- **Husband_Occupation** - Husband's occupation (categorical) 1, 2, 3, 4(random)
- **Standard_of_living_index** - Standard-of-living index (categorical) 1=verlow, 2, 3, 4=high
- **Media_exposure** - Media exposure (binary) Good, Not good
- **Contraceptive_method_used** - Contraceptive method used (class attribute) No,Yes

DATA INITIALISATION & PREPROCESSING

Data was initialized from given file and to verify that the data has been properly imported we look at the five head values of the data i.e., the first five rows.

	Wife_age	Wife_ education	Husband_education	No_of_children_born	Wife_religion	Wife_Working	Husband_Occupation	Standard_of_living_index	Media_exposure	Contraceptive_method_used
0	24.0	Primary	Secondary	3.0	Scientology	No	2	High	Exposed	No
1	45.0	Uneducated	Secondary	10.0	Scientology	No	3	Very High	Exposed	No
2	43.0	Primary	Secondary	7.0	Scientology	No	3	Very High	Exposed	No
3	42.0	Secondary	Primary	9.0	Scientology	No	3	High	Exposed	No
4	36.0	Secondary	Secondary	8.0	Scientology	No	3	Low	Exposed	No

Figure 31
First 5 values of dataframe

It appears as if everything is loaded properly onto a dataframe. Now checking the overall size of the data, we get:

```
No. of Rows: 1473
No. of Columns: 10
```

Figure 32
Size of dataframe.

It appears that the data has 1473 rows and 10 columns. Checking the data types of these columns:

```
RangeIndex: 1473 entries, 0 to 1472
Data columns (total 10 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   Wife_age                  1402 non-null   float64
 1   Wife_ education           1473 non-null   object
 2   Husband_education         1473 non-null   object
 3   No_of_children_born       1452 non-null   float64
 4   Wife_religion             1473 non-null   object
 5   Wife_Working              1473 non-null   object
 6   Husband_Occupation        1473 non-null   int64
 7   Standard_of_living_index  1473 non-null   object
 8   Media_exposure            1473 non-null   object
 9   Contraceptive_method_used 1473 non-null   object
dtypes: float64(2), int64(1), object(7)
```

Figure 33
Data type summary

Looking at the datatype info we can clearly see that there are a few missing values in certain columns and let us have a look at this in more detail:

```
Wife_age                    71
Wife_ education              0
Husband_education            0
No_of_children_born         21
Wife_religion                0
Wife_Working                 0
Husband_Occupation           0
Standard_of_living_index     0
Media_exposure               0
Contraceptive_method_used    0
```

Figure 34
Null values found in dataframe.

It appears the columns 'Wife_age' and 'No_of_children_born' have missing values. Before we perform any kind of data preprocessing let us first have a look at the 5-point summary:

	count	unique	top	freq	mean	std	min	25%	50%	75%	max
Wife_age	1402.0	NaN	NaN	NaN	32.606277	8.274927	16.0	26.0	32.0	39.0	49.0
Wife_ education	1473	4	Tertiary	577	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Husband_education	1473	4	Tertiary	899	NaN	NaN	NaN	NaN	NaN	NaN	NaN
No_of_children_born	1452.0	NaN	NaN	NaN	3.254132	2.365212	0.0	1.0	3.0	4.0	16.0
Wife_religion	1473	2	Scientology	1253	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Wife_Working	1473	2	No	1104	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Husband_Occupation	1473.0	NaN	NaN	NaN	2.137814	0.864857	1.0	1.0	2.0	3.0	4.0
Standard_of_living_index	1473	4	Very High	684	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Media_exposure	1473	2	Exposed	1364	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Contraceptive_method_used	1473	2	Yes	844	NaN	NaN	NaN	NaN	NaN	NaN	NaN

Figure 35
5 Point summary before pre processing.

'Wife_age' is imputed with its respective **median value** and 'No_of_children_born' is imputed with its respective **mode value.**

Now checking for Zero values:

```
Wife_age                    0
Wife_ education              0
Husband_education            0
No_of_children_born         97
Wife_religion                0
Wife_Working                 0
Husband_Occupation           0
Standard_of_living_index     0
Media_exposure               0
Contraceptive_method_used    0
```

Figure 36
Zero '0' values found

Having '0' values in 'No_of_children_born' is not an invalid entry so we shall not perform any preprocessing with respect to that.

The program returns a positive for duplicated values, but these values are not duplicated they are unique. Hence, no preprocessing will be done in that regard (refer notebook for a better image).

	Wife_age	Wife_ education	Husband_education	No_of_children_born	Wife_religion	Wife_Working	Husband_Occupation	Standard_of_living_index	Media_exposure	Contraceptive_method_used
79	38.0	Tertiary	Tertiary	1.0	Scientology	Yes	1	Very High	Exposed	No
167	26.0	Tertiary	Tertiary	1.0	Scientology	No	1	Very High	Exposed	No
224	47.0	Tertiary	Tertiary	4.0	Scientology	No	1	Very High	Exposed	No
270	30.0	Tertiary	Tertiary	2.0	Scientology	No	1	Very High	Exposed	No
299	26.0	Tertiary	Tertiary	1.0	Scientology	No	1	Very High	Exposed	No
394	29.0	Tertiary	Tertiary	0.0	Scientology	Yes	2	Very High	Exposed	No
414	20.0	Primary	Secondary	3.0	Scientology	No	3	Very High	Exposed	No
462	36.0	Tertiary	Tertiary	3.0	Scientology	No	1	Very High	Exposed	Yes
492	37.0	Tertiary	Tertiary	3.0	Scientology	No	1	Very High	Exposed	Yes
528	29.0	Tertiary	Tertiary	2.0	Scientology	Yes	1	High	Exposed	Yes
576	41.0	Tertiary	Tertiary	4.0	Non-Scientology	Yes	2	Very High	Exposed	Yes
585	39.0	Tertiary	Tertiary	3.0	Scientology	No	1	Very High	Exposed	Yes
586	24.0	Tertiary	Tertiary	1.0	Scientology	No	1	Very High	Exposed	Yes
622	46.0	Tertiary	Tertiary	4.0	Scientology	No	1	Very High	Exposed	Yes
627	44.0	Tertiary	Tertiary	4.0	Scientology	No	1	Very High	Exposed	Yes
646	24.0	Tertiary	Tertiary	1.0	Scientology	Yes	1	Very High	Exposed	Yes
655	29.0	Tertiary	Tertiary	2.0	Scientology	No	3	Very High	Exposed	Yes
682	35.0	Tertiary	Tertiary	4.0	Scientology	No	1	Very High	Exposed	Yes
688	36.0	Tertiary	Tertiary	4.0	Non-Scientology	No	1	Very High	Exposed	Yes
694	26.0	Primary	Tertiary	3.0	Scientology	No	3	Very High	Exposed	Yes
717	43.0	Tertiary	Tertiary	3.0	Scientology	No	1	Very High	Exposed	Yes
720	32.0	Tertiary	Tertiary	3.0	Non-Scientology	No	1	Very High	Exposed	Yes
721	46.0	Tertiary	Tertiary	4.0	Scientology	No	1	Very High	Exposed	Yes
738	32.0	Tertiary	Tertiary	2.0	Non-Scientology	Yes	1	Very High	Exposed	Yes
755	35.0	Tertiary	Tertiary	4.0	Scientology	No	1	Very High	Exposed	Yes
769	34.0	Tertiary	Tertiary	3.0	Scientology	No	1	Very High	Exposed	Yes
790	34.0	Tertiary	Tertiary	2.0	Scientology	No	1	Very High	Exposed	Yes
799	37.0	Tertiary	Tertiary	3.0	Scientology	No	2	Very High	Exposed	Yes
806	35.0	Tertiary	Tertiary	2.0	Scientology	Yes	1	Very High	Exposed	Yes
833	21.0	Secondary	Secondary	1.0	Scientology	No	3	High	Exposed	Yes

Figure 37
Duplicated values according to program (False Positive)

Now let us Check for outliers in the data:

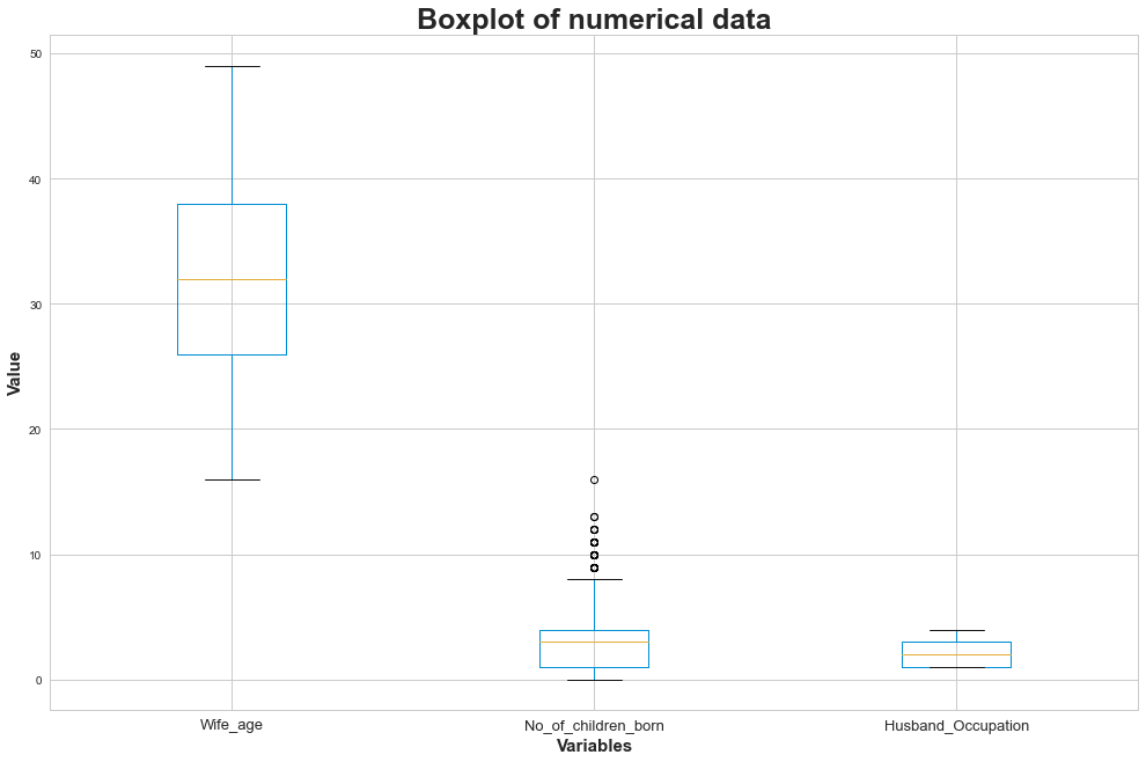

Figure 38 *Boxplot of numerical data to detect outliers*

Having outliers can have a large effect on the output we have no choice but to treat them.

we will proceed with removing outliers by moving the outliers to its closest quartile.
The result after removing outliers:

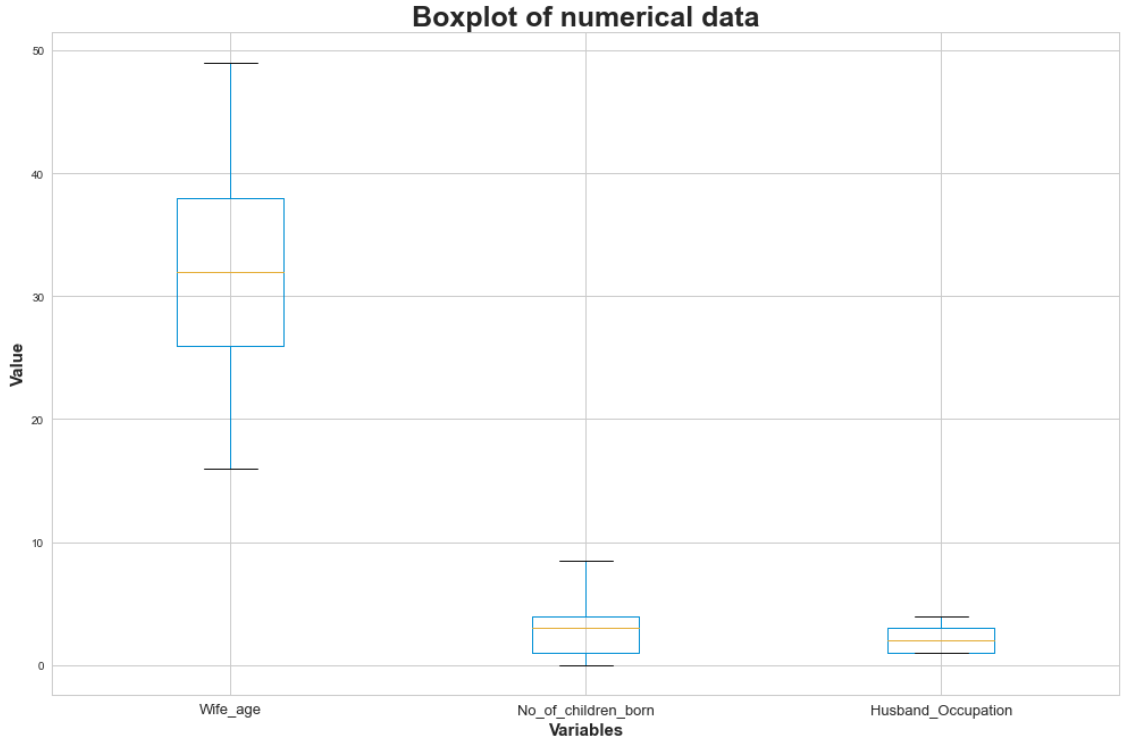

Figure 39 *Box plot of numerical data after outlier treatment*

The 5-point summary after preprocessing can be given as:

	count	unique	top	freq	mean	std	min	25%	50%	75%	max
Wife_age	1473.0	NaN	NaN	NaN	32.577054	8.073941	16.0	26.0	32.0	38.0	49.0
Wife_ education	1473	4	Tertiary	577	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Husband_education	1473	4	Tertiary	899	NaN	NaN	NaN	NaN	NaN	NaN	NaN
No_of_children_born	1473.0	NaN	NaN	NaN	3.180244	2.185892	0.0	1.0	3.0	4.0	8.5
Wife_religion	1473	2	Scientology	1253	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Wife_Working	1473	2	No	1104	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Husband_Occupation	1473.0	NaN	NaN	NaN	2.137814	0.864857	1.0	1.0	2.0	3.0	4.0
Standard_of_living_index	1473	4	Very High	684	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Media_exposure	1473	2	Exposed	1364	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Contraceptive_method_used	1473	2	Yes	844	NaN	NaN	NaN	NaN	NaN	NaN	NaN

Figure 40
5 point data summary after preprocessing

EXPLORATORY DATA ANALYSIS (EDA)

Now that all the required preprocessing has completed, we can briefly explore the data to find any interesting trends or insights.

The most important aspect that we have to explore is the relationship between the numerical variables as that is what we would be most interested in with respect to linear regression.

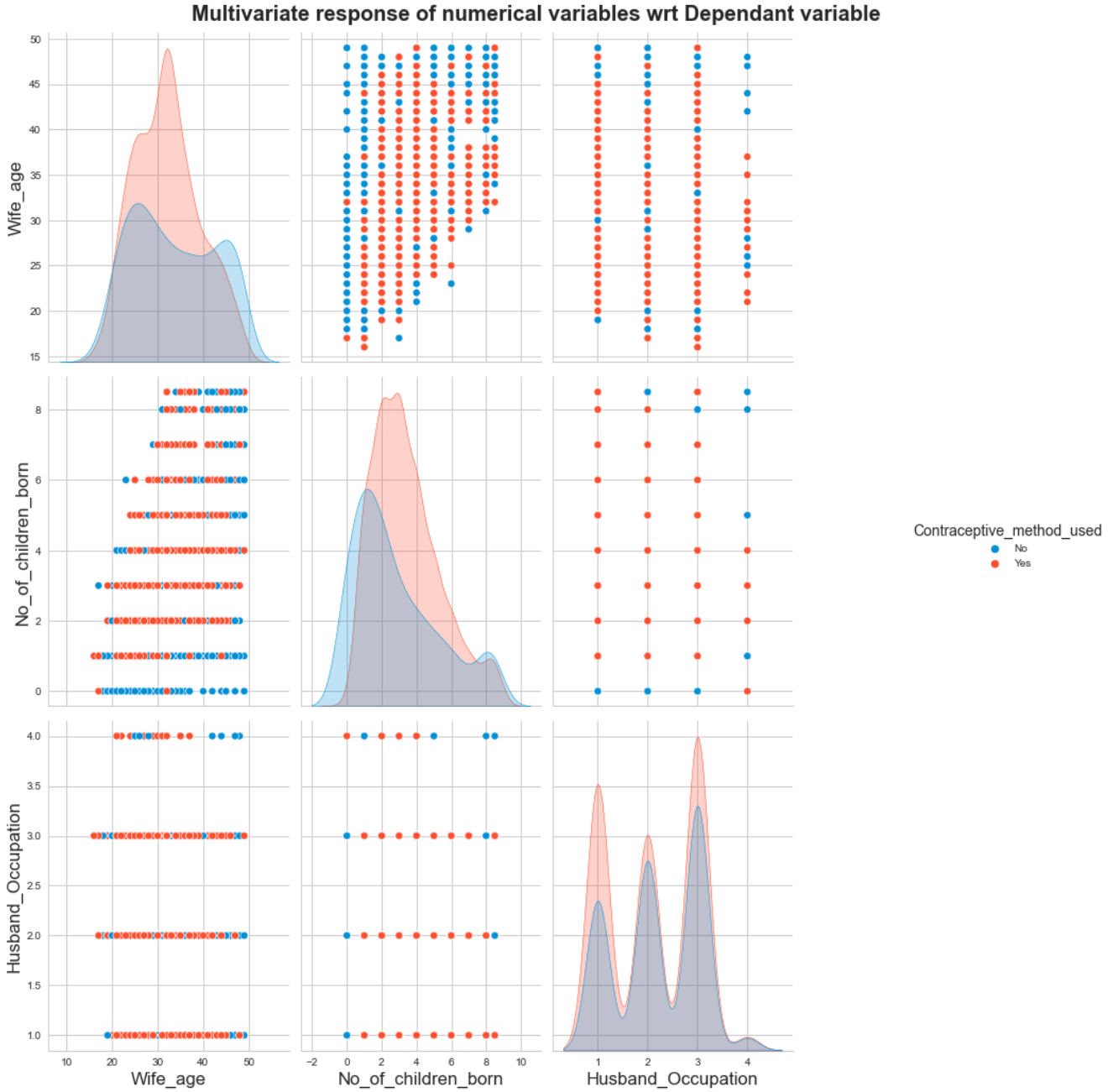

Figure 41

pairplot of numerical data showing bi variate plots and dist.

The corresponding Heatmap is given as:

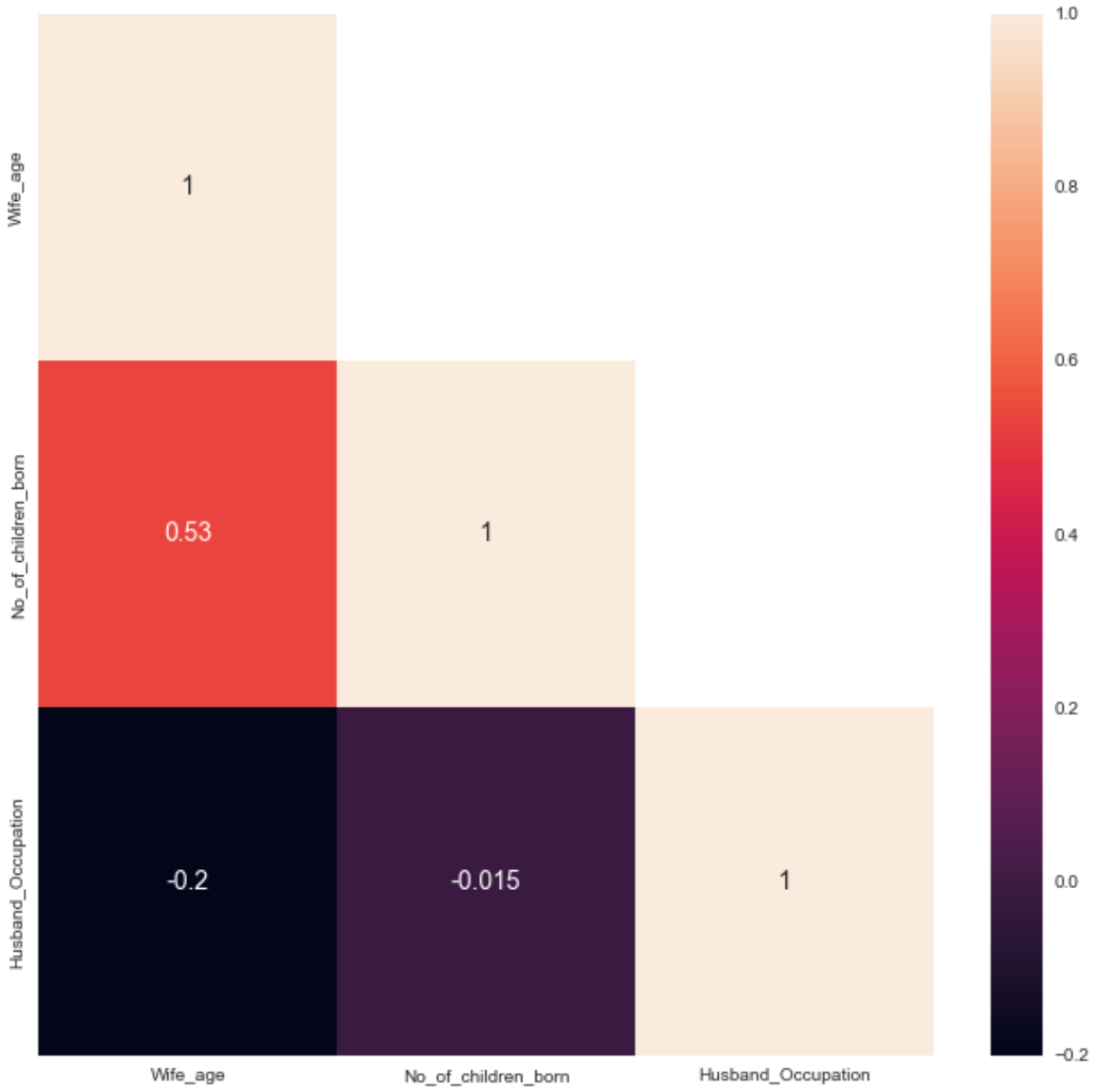

Figure 42

Heatmap of Correlation of numerical Variables

There appears a strong correlation between 'wife_age' and 'No_of_children_born' which makes sense that it has a positive correlation.

Before we proceed with regression, we split the data(random_state=0) in a 70:30 ratio between train and test data.

The data split happened as shown:

```
Number of rows and columns of the training set for the independent variables: (1031, 15)
Number of rows and columns of the training set for the dependent variable: (1031,)
Number of rows and columns of the test set for the independent variables: (442, 15)
Number of rows and columns of the test set for the dependent variable: (442,)
```

The dependent variable appears to be split in the same ratio in both train and test variable are required.

Table 7: *Table depicting split of dependent variable outputs in train and test data.*

Y	Train	Test
0	0.42677	0.427602
1	0.57323	0.572398

LOGISTIC REGRESSION

After setting up the train and test data we create a logistic regression model with the parameters stated below and fit the model on to our data.

```
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 16 concurrent workers.
[Parallel(n_jobs=-1)]: Done   1 out of   1 | elapsed:    0.2s finished
LogisticRegression(max_iter=10000, n_jobs=-1, penalty='none',
                   solver='newton-cg', verbose=True)
```

Figure 43
Logistic regression Parameters and model fit prompt.

After application of the logistic regression model the accuracy scores obtained are as follows:

```
The Accuracy score (Train data) is 0.677
The Accuracy score (Test data) is 0.6833
```

Figure 44 *Accuracy Score - Logistic Regression*

And the AUC for that same model is also given as:

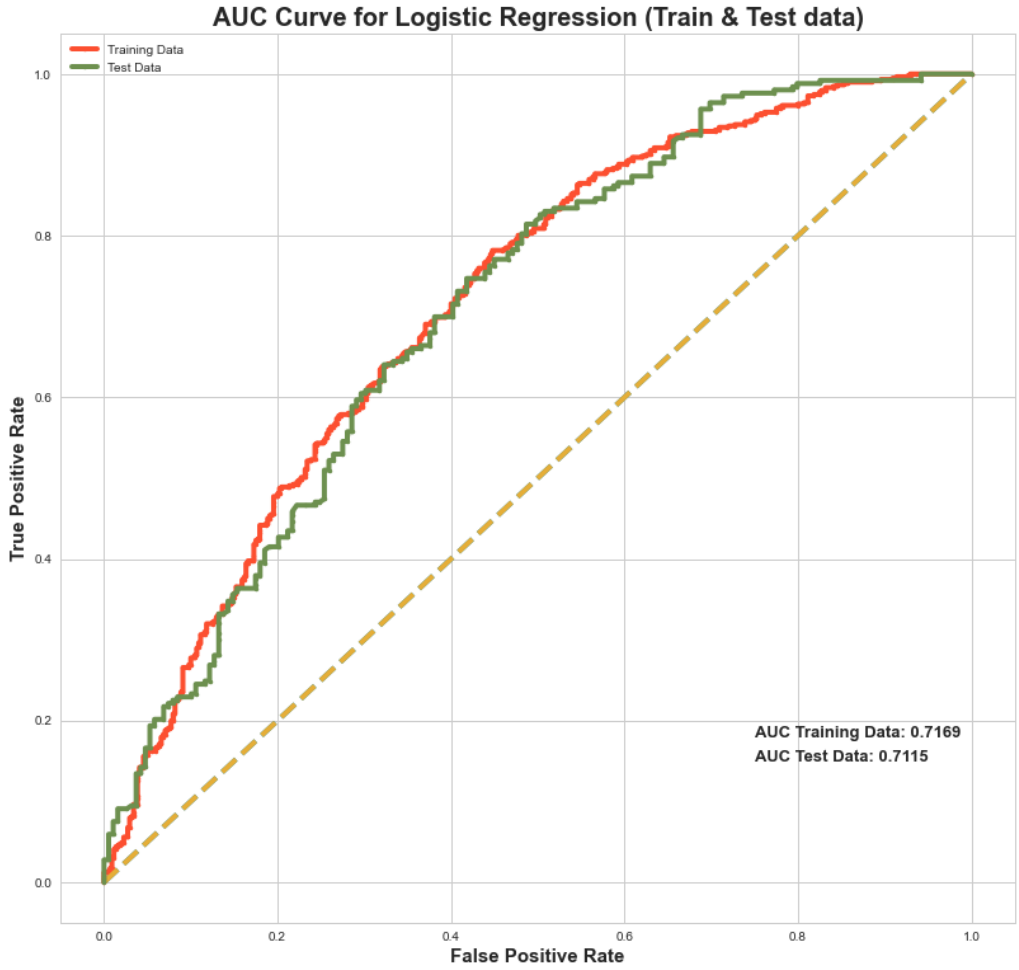

Figure 45
AUC curve and score for Logistic regression

The confusion matrix of the fit logistic regression model:

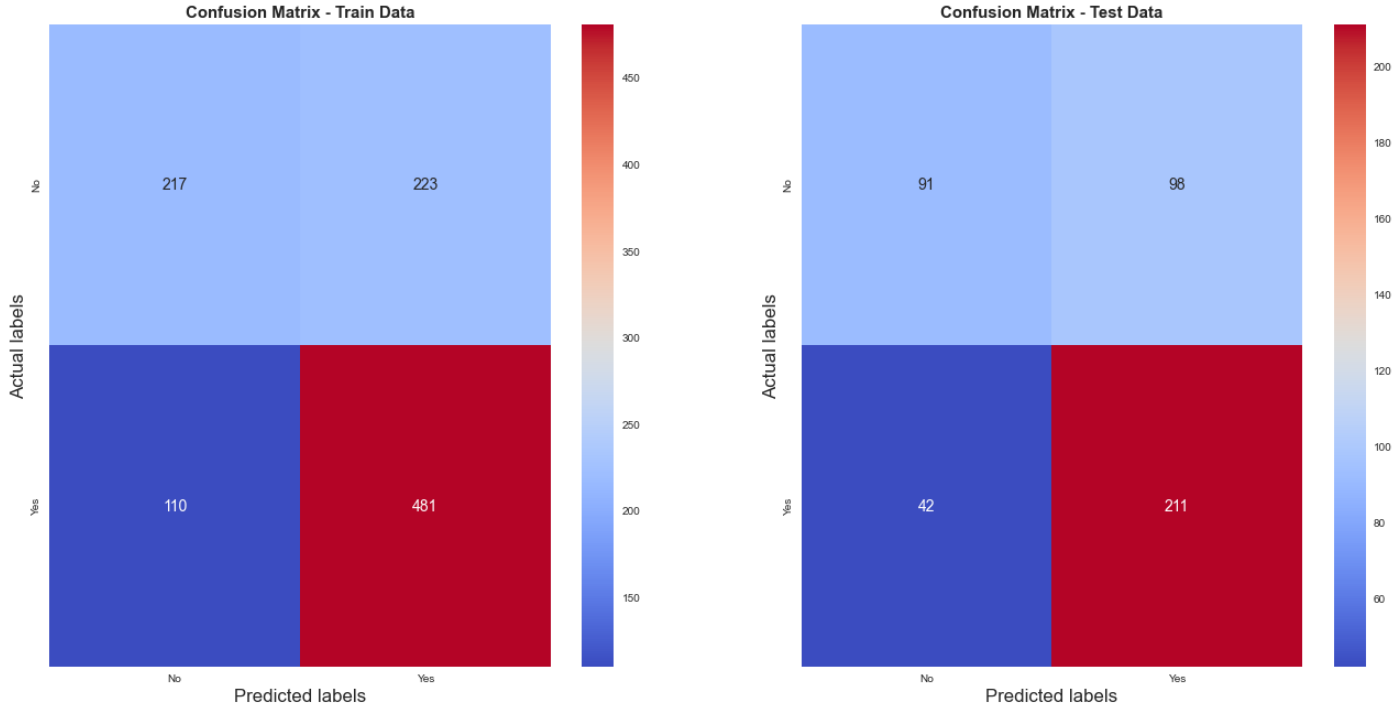

Figure 46 *Confusion Matrix – Logistic Regression*

And the resultant classification report giving the overall summary:

Classification Report of the training data:

	precision	recall	f1-score	support
0	0.6636	0.4932	0.5658	440
1	0.6832	0.8139	0.7429	591
accuracy			0.6770	1031
macro avg	0.6734	0.6535	0.6543	1031
weighted avg	0.6749	0.6770	0.6673	1031

Classification Report of the test data:

	precision	recall	f1-score	support
0	0.6842	0.4815	0.5652	189
1	0.6828	0.8340	0.7509	253
accuracy			0.6833	442
macro avg	0.6835	0.6577	0.6581	442
weighted avg	0.6834	0.6833	0.6715	442

Figure 47
Classification Report – Logistic Regression

Based on all the metric gathered above it seems that the model does not have a problem of overfitting or underfitting. We shall compare the rest of the metrics to the other models which we will be utilizing and summarize it at a later point.

LINEAR DISCRIMINANT ANALYSIS (LDA)

After setting up the train and test data we create a LDA model and fit the model on to our data. After application of the LDA model (Threshold 0.5) the accuracy scores obtained are as follows:

```
The Accuracy score (Train data) is 0.676
The Accuracy score (Test data) is 0.6742
```

Figure 48 *Accuracy Score - LDA*

And the AUC for that same model is also given as:

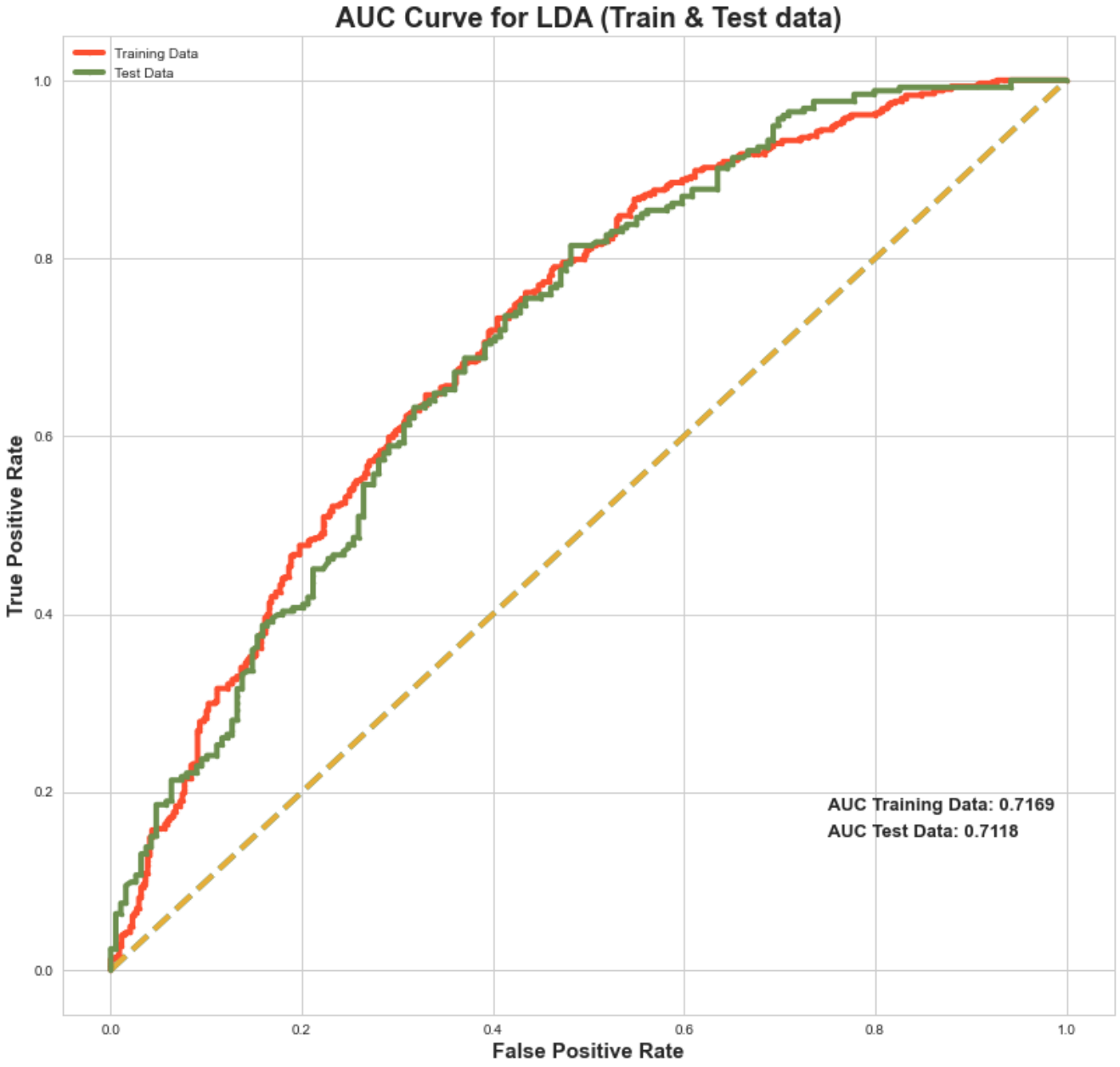

Figure 49
AUC curve and score for LDA

The confusion matrix of the fit LDA model:

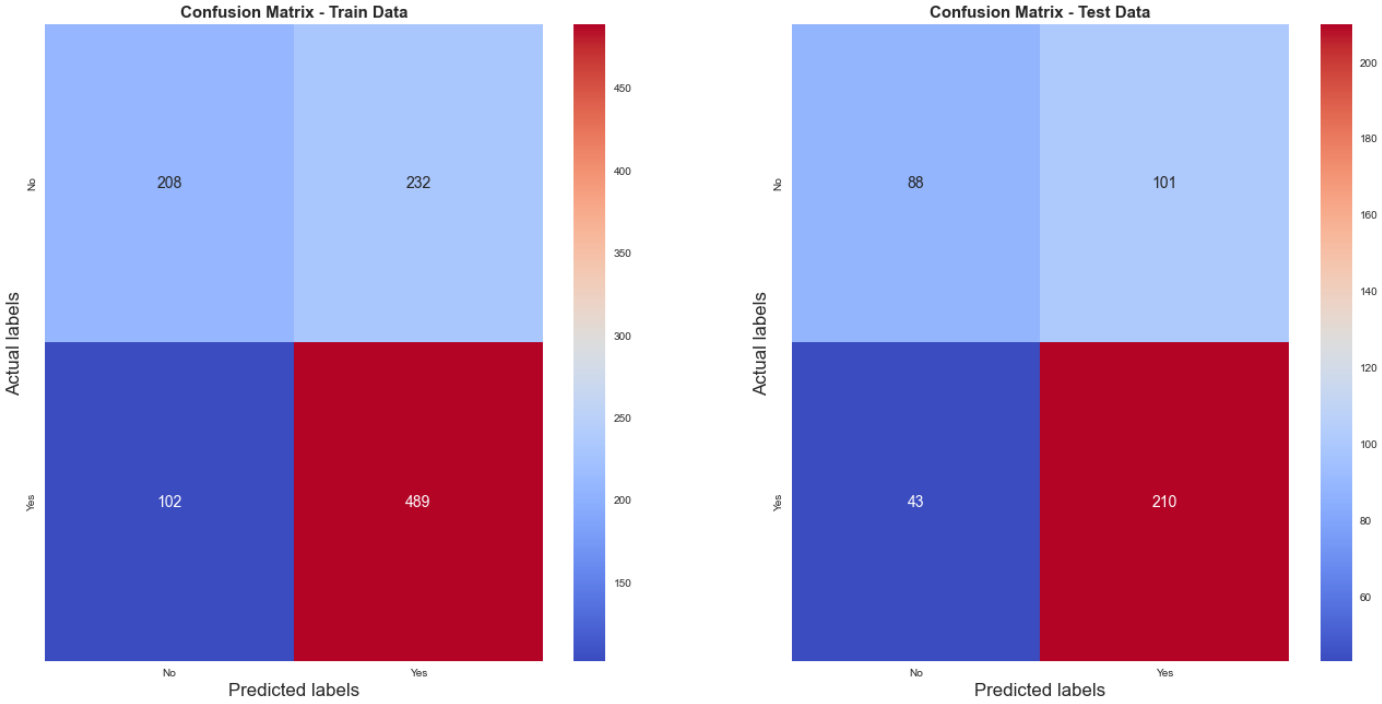

Figure 50 *Confusion Matrix – LDA*

And the resultant classification report giving the overall summary:

Classification Report of the training data:

	precision	recall	f1-score	support
0	0.6710	0.4727	0.5547	440
1	0.6782	0.8274	0.7454	591
accuracy			0.6760	1031
macro avg	0.6746	0.6501	0.6500	1031
weighted avg	0.6751	0.6760	0.6640	1031

Classification Report of the test data:

	precision	recall	f1-score	support
0	0.6718	0.4656	0.5500	189
1	0.6752	0.8300	0.7447	253
accuracy			0.6742	442
macro avg	0.6735	0.6478	0.6473	442
weighted avg	0.6738	0.6742	0.6614	442

Figure 51 *Classification Report – LDA*

We also check if changing threshold values end up affecting our metrics:

```
Cutoff: 0.1     Accuracy: 0.5887 Recall: 1.0 Precision: 0.5823 F1 Score: 0.736
Cutoff: 0.2     Accuracy: 0.6091 Recall: 0.9932 Precision: 0.5953 F1 Score: 0.7445
Cutoff: 0.3     Accuracy: 0.6353 Recall: 0.9679 Precision: 0.6157 F1 Score: 0.7526
Cutoff: 0.4     Accuracy: 0.6712 Recall: 0.9154 Precision: 0.6518 F1 Score: 0.7614
Cutoff: 0.5     Accuracy: 0.676 Recall: 0.8274 Precision: 0.6782 F1 Score: 0.7454
Cutoff: 0.6     Accuracy: 0.6499 Recall: 0.6565 Precision: 0.7106 F1 Score: 0.6825
Cutoff: 0.7     Accuracy: 0.5955 Recall: 0.4196 Precision: 0.7702 F1 Score: 0.5433
Cutoff: 0.8     Accuracy: 0.4985 Recall: 0.1574 Precision: 0.8304 F1 Score: 0.2646
Cutoff: 0.9     Accuracy: 0.4326 Recall: 0.0102 Precision: 1.0 F1 Score: 0.0201
```

Figure 52
Accuracy for different thresholds

Graphically summarized as:

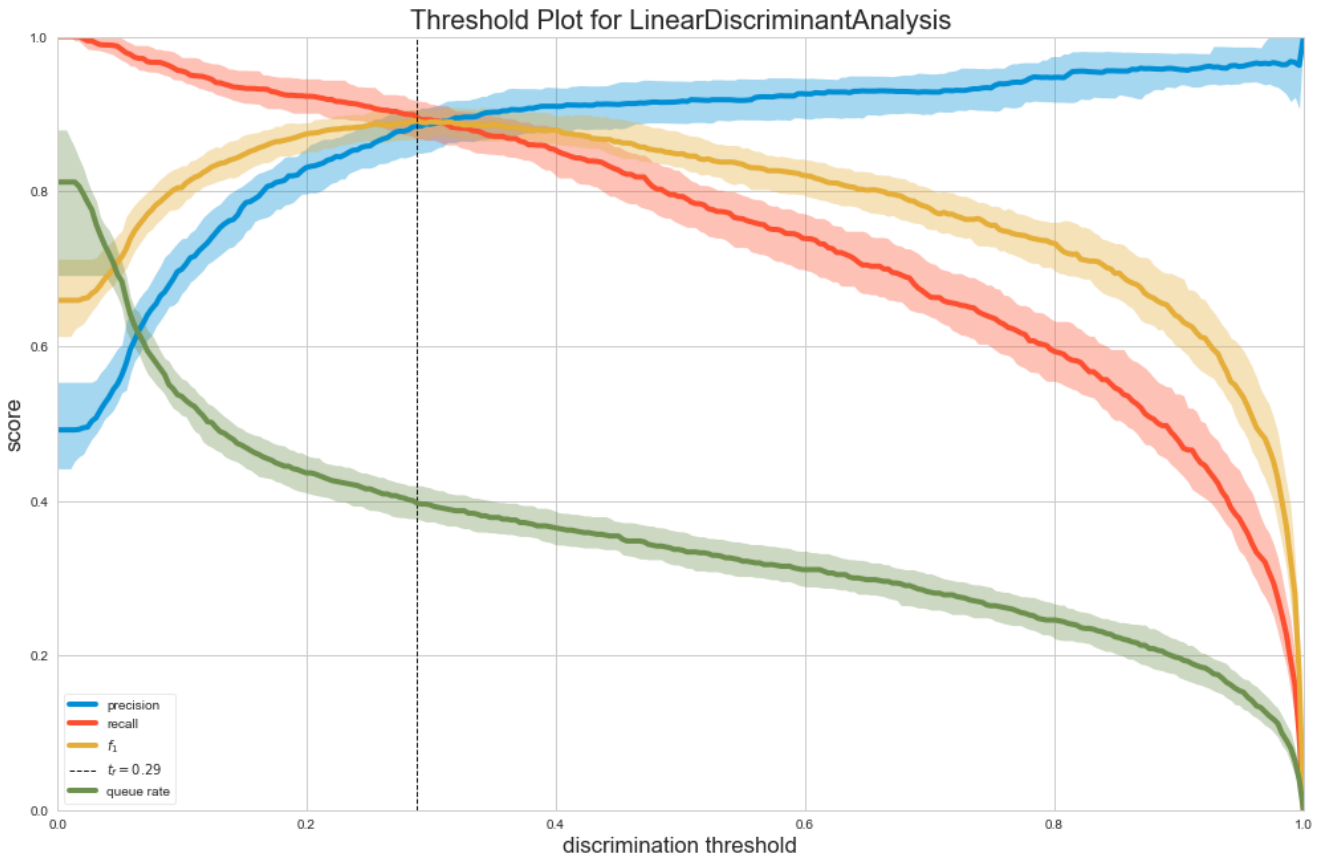

Figure 53
Threshold Plot

Based on all the metric gathered above it seems that the model does not have a problem of overfitting or underfitting. Also, it appears that the **highest accuracy** appears to be for the original default value of **threshold = 0.5**.

We shall compare the rest of the metrics to the other models which we will be utilizing and summarize it after we finish the CART Model.

CART MODEL

After setting up the train and test data appropriately for CART models we proceed to create a Decision Tree model and fit the model on to our data. After application of the Decision Tree model the tree obtained (without any pruning) are as follow:

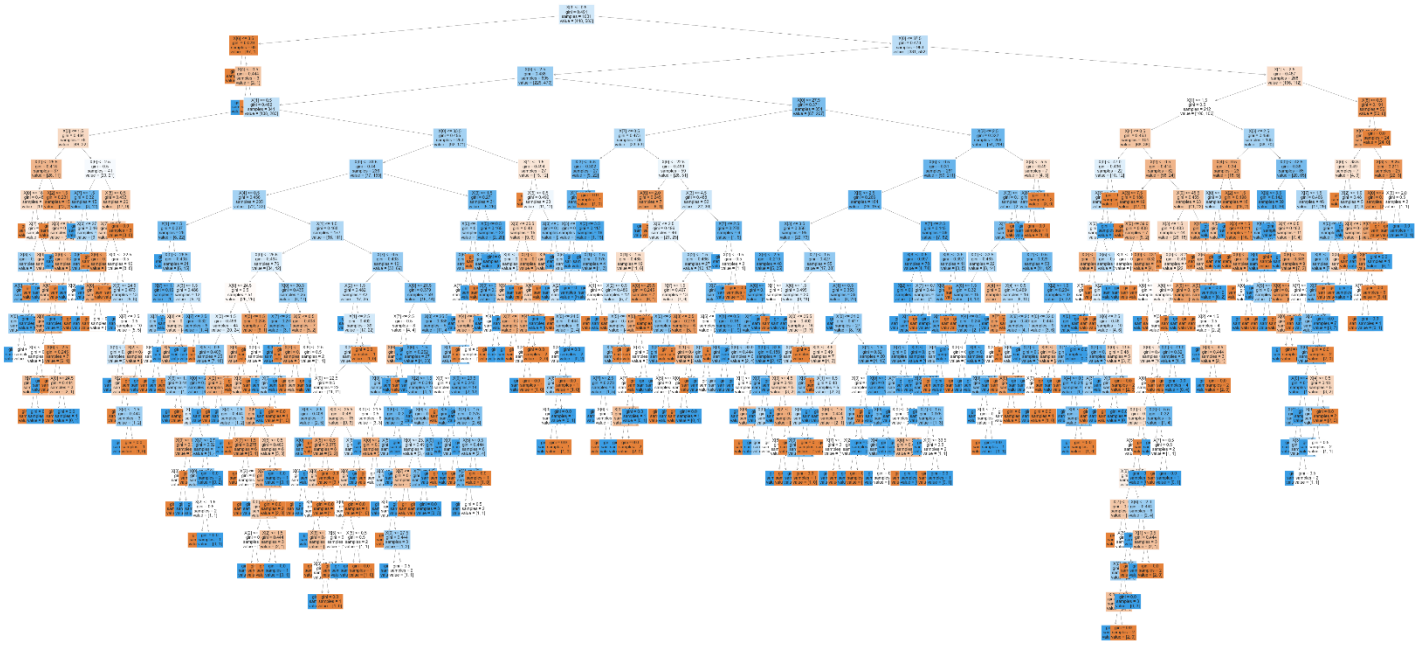

Figure 54 *Decision tree - I*

And the important features of the decision tree are given as:

	Features	Importance
3	No_of_children_born	0.469
0	Wife_age	0.241
1	Wife_ education	0.216
4	Wife_religion	0.025
7	Standard_of_living_index	0.022
2	Husband_education	0.014
8	Media_exposure	0.007
6	Husband_Occupation	0.005
5	Wife_Working	0.000

Figure 55 *Feature Importance - I*

This can be visualized as:

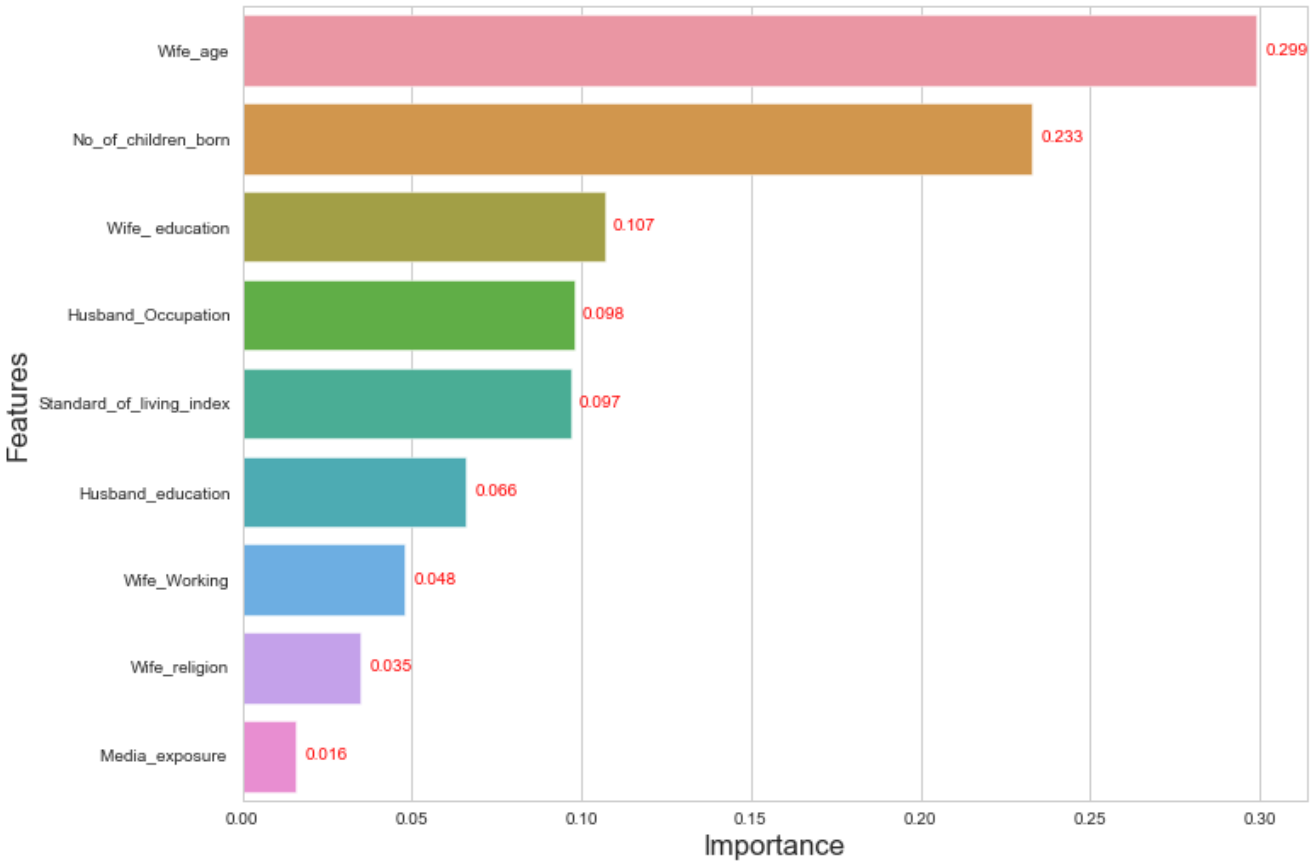

Figure 56 *Visualization of Feature Importance - I*

Obviously, this decision tree is not optimized as we can clearly see that the depth is too high, and the no. of leaves also are too high while some only containing one sample at most. Therefore, we proceed to optimize the model and prune the decision tree for the best possible accuracy score and the best possible fit.

The result is as follows:

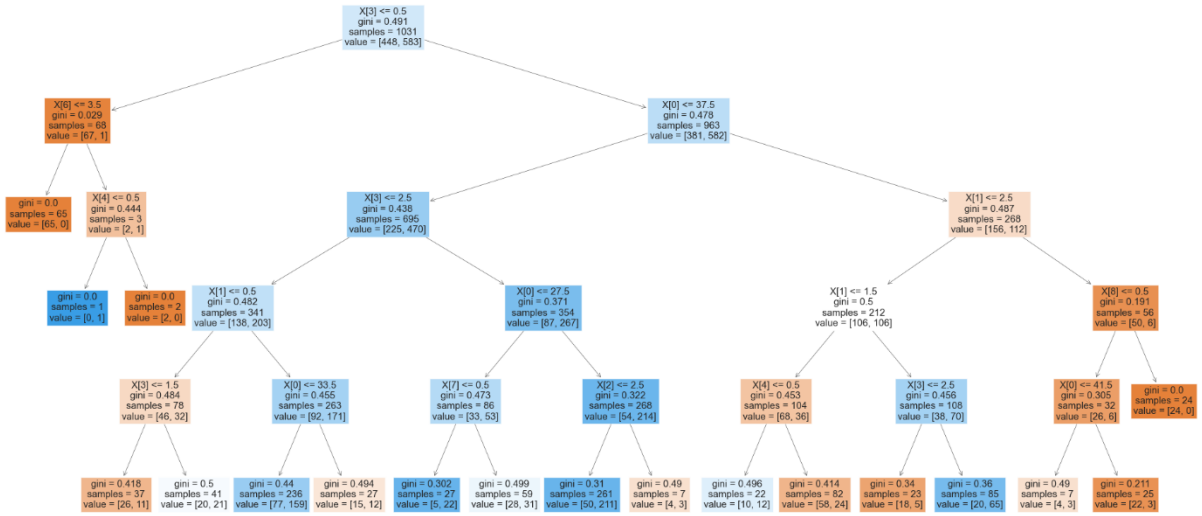

Figure 57 *Decision tree - II*

And the important features of the decision tree are given as:

	Features	Importance
3	No_of_children_born	0.469
0	Wife_age	0.241
1	Wife_ education	0.216
4	Wife_religion	0.025
7	Standard_of_living_index	0.022
2	Husband_education	0.014
8	Media_exposure	0.007
6	Husband_Occupation	0.005
5	Wife_Working	0.000

Figure 58 *Feature Importance - II*

This can be visualized as:

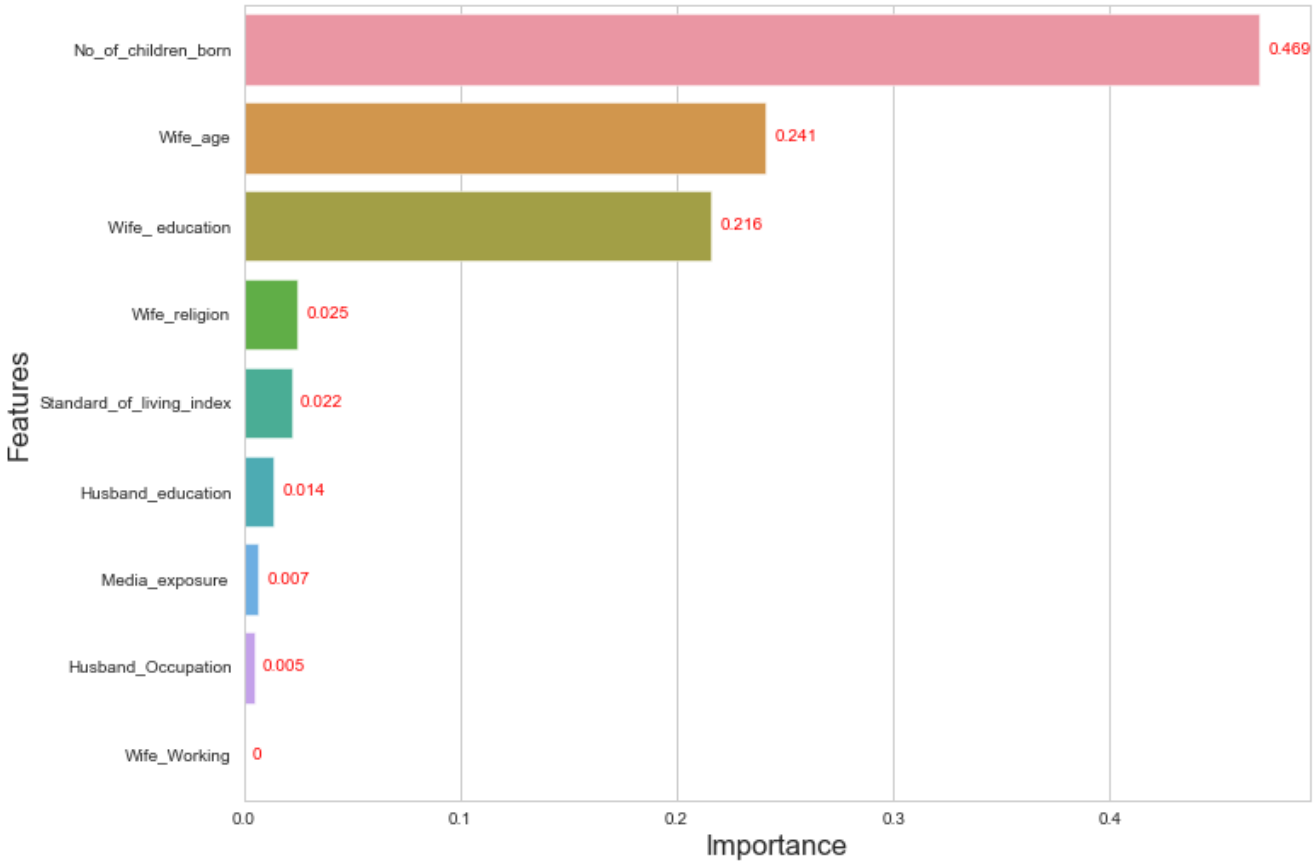

Figure 59 *Visualization of Feature Importance - II*

The Accuracy Score is given as:

```
The Accuracy score (Train data) is 0.7371
The Accuracy score (Test data) is 0.7195
```

Figure 60 *Accuracy Score - CART*

And the AUC for that same model is also given as:

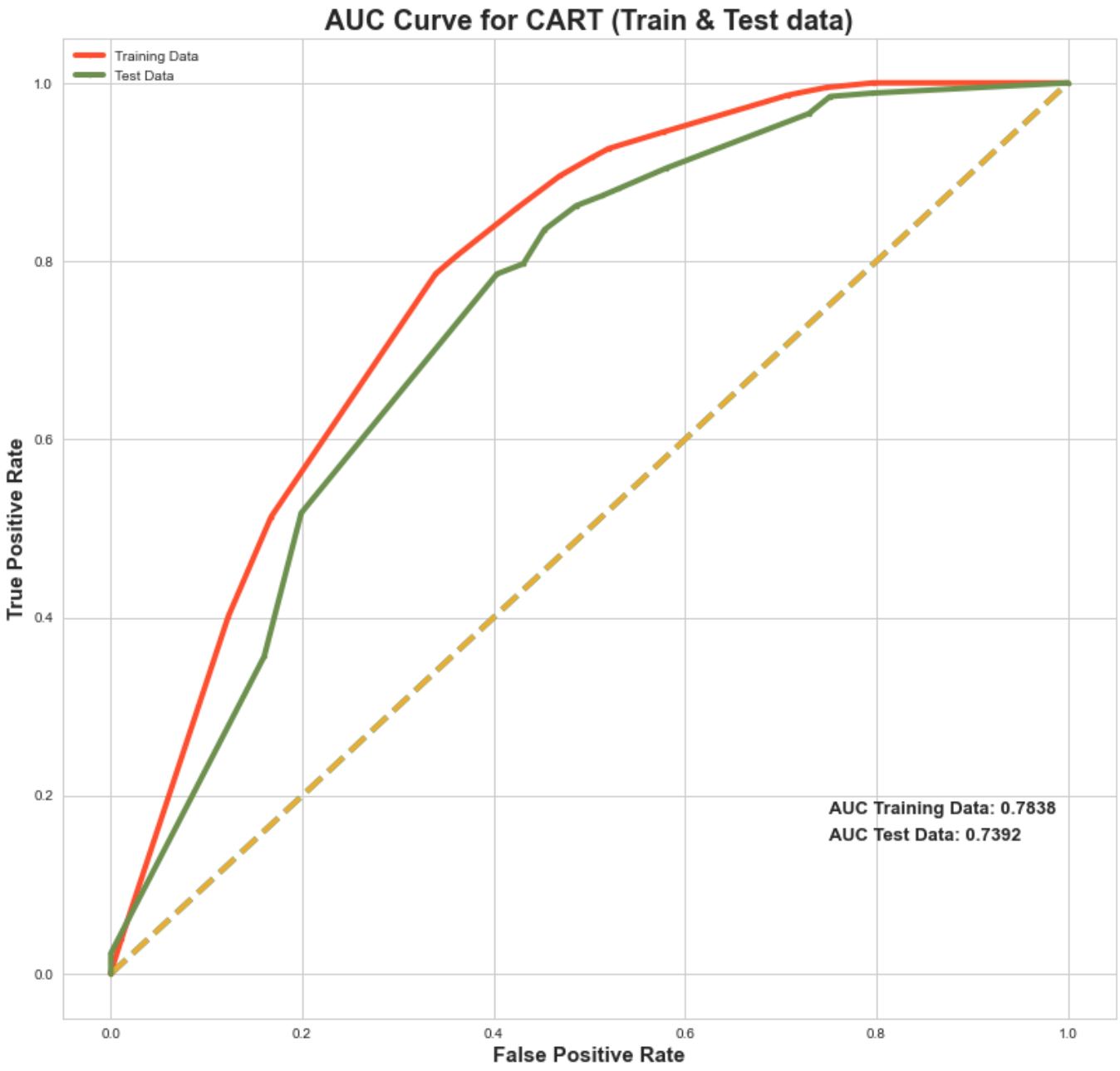

Figure 61
AUC Curve and AUC Score - CART

The confusion matrix of the regularized CART model:

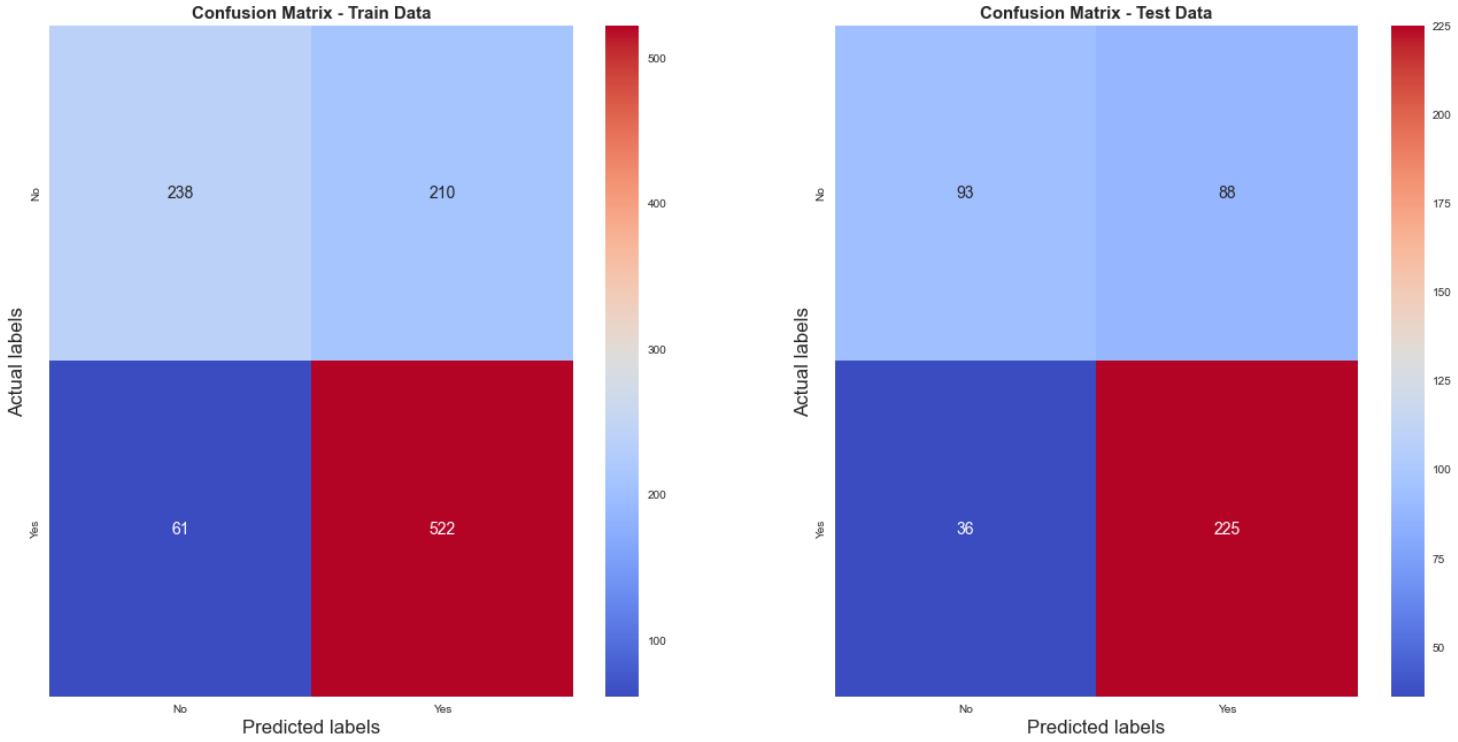

Figure 62 *Confusion Matrix - CART*

And the resultant classification report giving the overall summary:

Classification Report of the training data:

	precision	recall	f1-score	support
0	0.7960	0.5312	0.6372	448
1	0.7131	0.8954	0.7939	583
accuracy			0.7371	1031
macro avg	0.7546	0.7133	0.7156	1031
weighted avg	0.7491	0.7371	0.7258	1031

Classification Report of the test data:

	precision	recall	f1-score	support
0	0.7209	0.5138	0.6000	181
1	0.7188	0.8621	0.7840	261
accuracy			0.7195	442
macro avg	0.7199	0.6879	0.6920	442
weighted avg	0.7197	0.7195	0.7086	442

Figure 63 *Classification Report - CART*

Based on all the metric gathered above it seems that the model does not have a problem of overfitting or underfitting. We shall compare all the models and their performance and overall insights next.

SUMMARY, INSIGHTS & RECOMMENDATIONS

- It would seem that the regularized Decision Tree CART model performed the best in terms of all parameters i.e., High accuracy and precision with no problems of under fitting or overfitting.

- Based on the models the most important factors that determine whether or not contraception is used is No. of Children already present, Wife's Age and Her Education.

- Whereas media exposure yields to little or no effect on the use of contraception.

- Therefore, in order to enhance the use of contraceptive measures efforts must be focused such that any and all resources related to contraceptives reach women who fit into the categories stated above.

- Another insight is that now we know the poorer performing factors efforts can me made to overhaul the strategy or campaign with respect to those factors.