FEBRUARY 5, 2023

Project - Predictive Modelling

YARESH VIJAYASUNDARAM
GREAT LEARNING
Post Graduate Program in Data Science and Business Analytics

Table of Contents

Problem 1**: Linear Regression**

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:

System measures used:

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

1.1 Read the data and do exploratory data analysis. Describe the data briefly. (Check the Data types, shape, EDA, 5 point summary). Perform Univariate, Bivariate Analysis, Multivariate Analysis.

The below fig. 1 shows the first five rows of the dataset:

	lread	lwrite	scall	sread	swrite	fork	exec	rchar	wchar	pgout	...	pgscan	atch	pgin	ppgin	pflt	vflt	runqsz	freemem	freeswap
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
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
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
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
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

5 rows × 22 columns

Figure 1: First five rows of the dataset

The below fig 2 shows the last five rows of the dataset:

	lread	lwrite	scall	sread	swrite	fork	exec	rchar	wchar	pgout	...	pgscan	atch	pgin	ppgin	pflt	vflt	runqsz	freemem	fre
8187	16	12	3009	360	244	1.6	5.81	405250.0	85282.0	8.02	...	55.11	0.6	35.87	47.90	139.28	270.74	CPU_Bound	387	9
8188	4	0	1596	170	146	2.4	1.80	89489.0	41764.0	3.80	...	0.20	0.8	3.80	4.40	122.40	212.60	Not_CPU_Bound	263	10
8189	16	5	3116	289	190	0.6	0.60	325948.0	52640.0	0.40	...	0.00	0.4	28.40	45.20	60.20	219.80	Not_CPU_Bound	400	9
8190	32	45	5180	254	179	1.2	1.20	62571.0	29505.0	1.40	...	18.04	0.4	23.05	24.25	93.19	202.81	CPU_Bound	141	10
8191	2	0	985	55	46	1.6	4.80	111111.0	22256.0	0.00	...	0.00	0.2	3.40	6.20	91.80	110.00	CPU_Bound	659	17

5 rows × 22 columns

Figure 2 Last five rows of the dataset

Data Information:

```
<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)
memory usage: 1.4+ MB
```

Figure 3: Data information

From fig 3, we can see there are **8192** rows and **22** columns. The Sun Sparcstation collects the dataset on computer systems activity measures. The dataset will be analysed to find a linear equation to build a model to predict 'usr'. There are 22 variables, out of which 13 are float data types, 8 are integer data types, and one is an object datatype. Furthermore, we could also in variables 'rchar' and 'wchar' having few missing values in the dataset.

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 4: 5 Point summary

The above fig 4 data description depicts the dataset's mean, median, min, max and lower and upper quartile values. It is interesting to see many variables, min and 25%, is zero in the datasets.

Univariate:

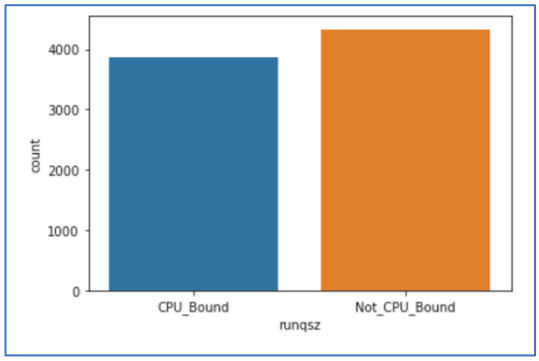


Figure 5: Univariate of runqsz variable

As we can see from the above fig 5, most activities are Not - CPU-bound tasks compared to the CPU-bound tasks- where 4331 tasks are not-CPU bound, and 3861 are CPU-Bound.

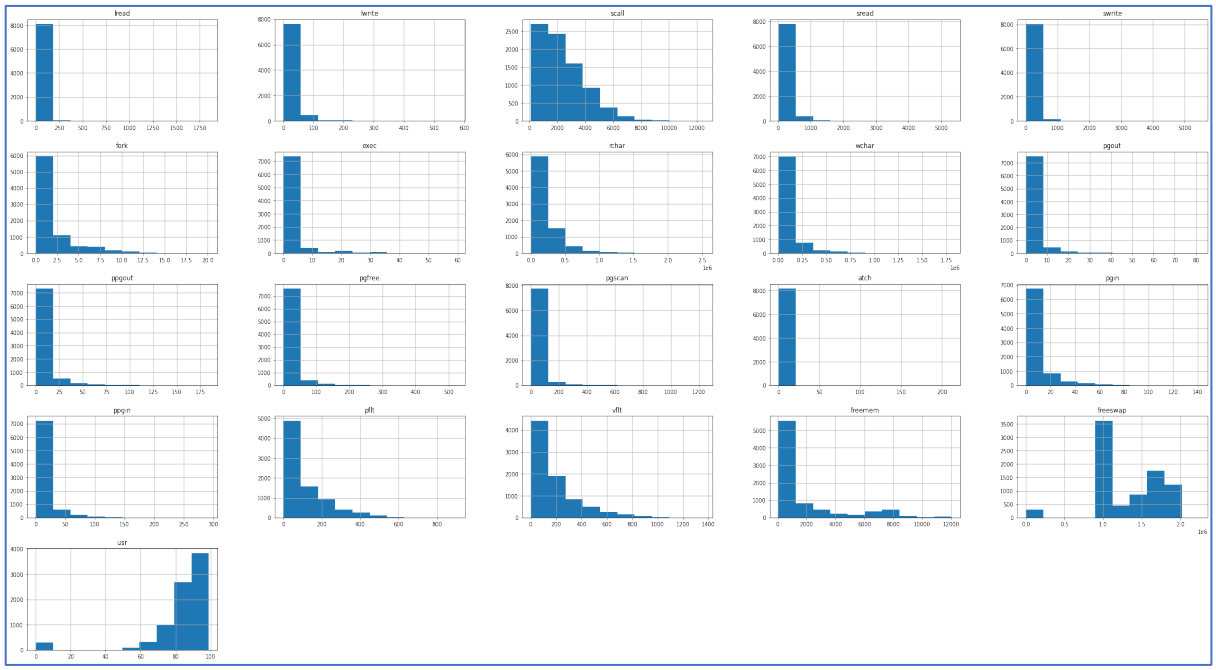


Figure 6: Histogram of all the variable

All the continuous variable has been plotted using the histogram to get an overall understanding of the features. It is evident from the above fig 6, that most of the variables are right–skewed and for which most of the values are zero or closer to zero except for the freeswap and usr feature.

Bivariate Analysis:

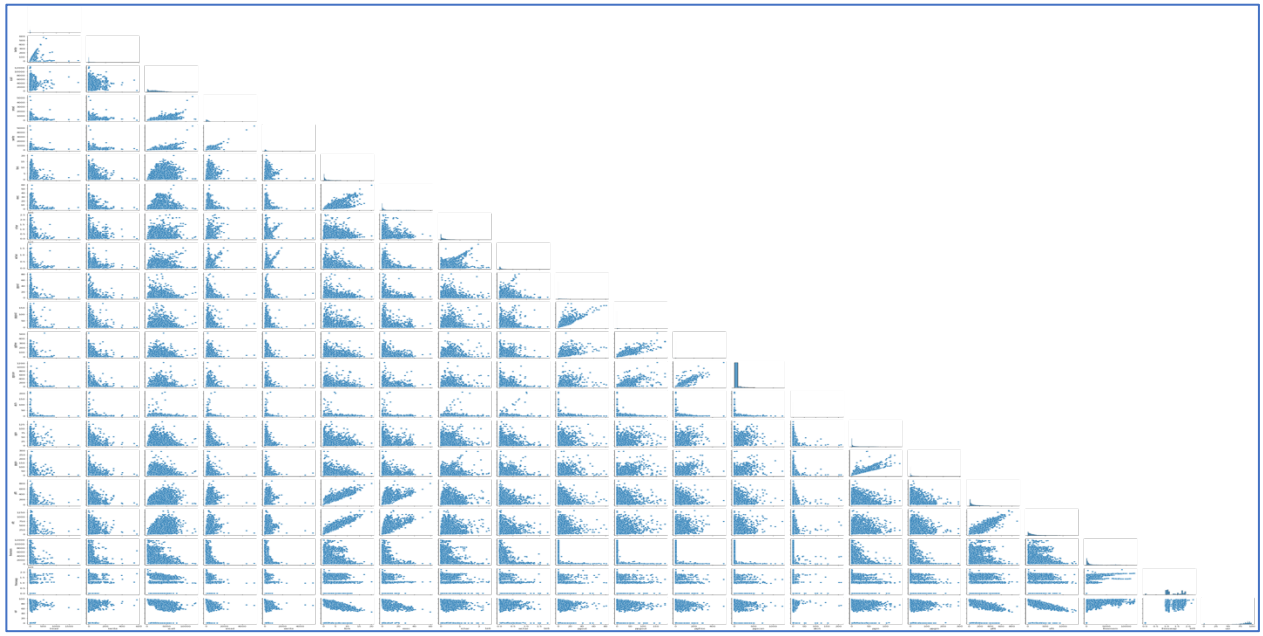


Figure 8: Scatter plot of all the features

From the above pair plot, it is clear that a good number of variables have a positive correlation, and most of the variables have a weak negative correlation when plotted against usr, freeswap and freemem.

Below are some plots that has a strong correlation:

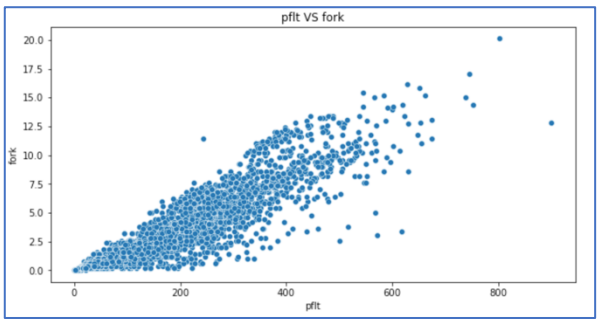


Figure 11: pflt vs fork

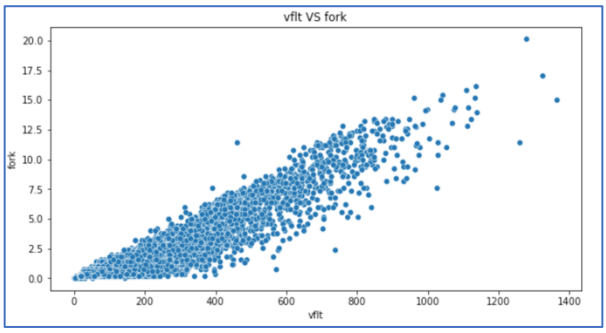


Figure 9: vflt vs fork

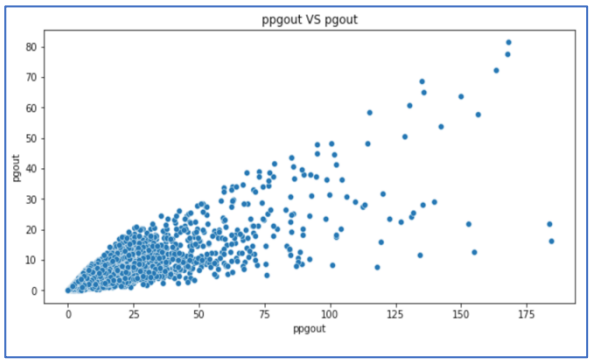


Figure 13: ppgout vs pgout

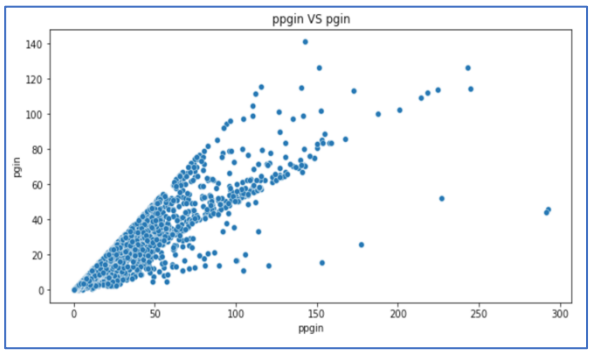


Figure 12: ppgin vs pgin

Heatmap of the dataset:

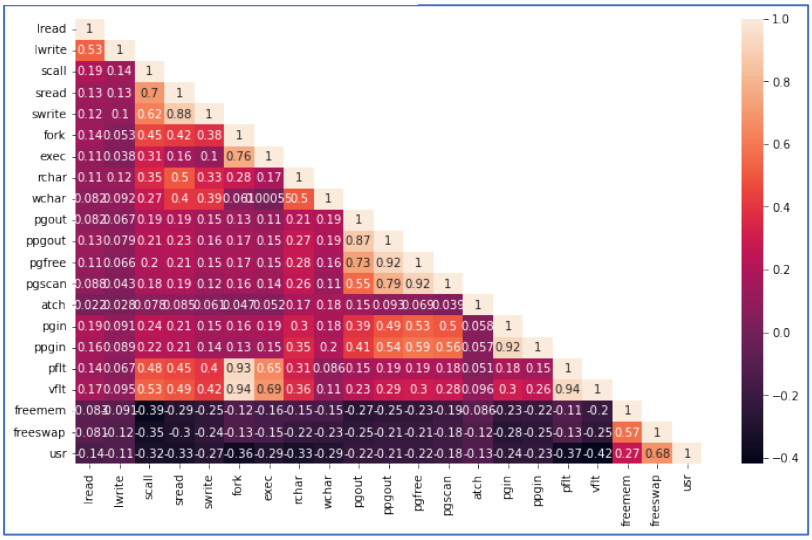


Figure 14: Heatmap

The above fig 12 represents the heatmap of the dataset, where we can see the strength of correlations between the variables in the data.

Multivariate Analysis:

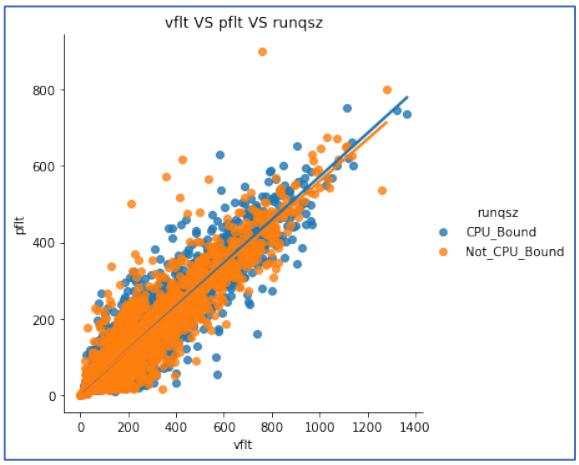


Figure 18: vflt vs pflt vs runqsz

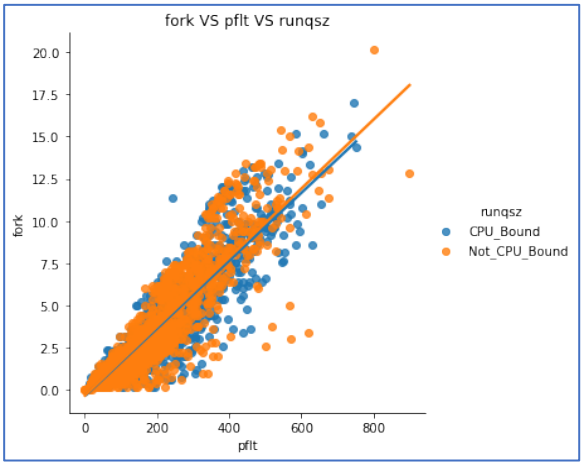


Figure 16: fork vs pflt vs runqsz

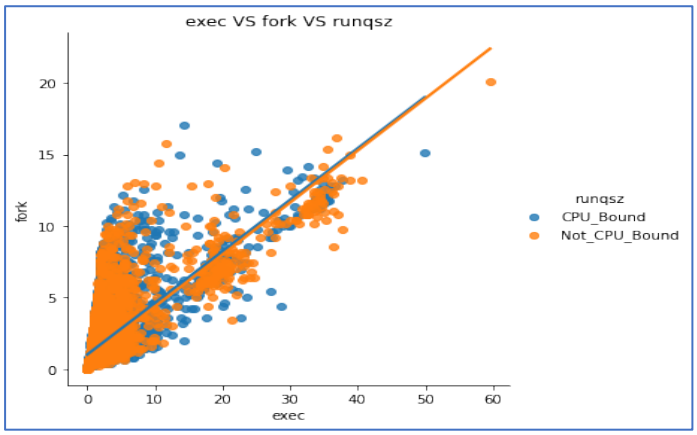


Figure 19: exec vs fork vs runqsz

From the above Fig 13, 14, & 15, it is evident that when we plot the continuous variables with the only categorical variable – 'runqsz'. There is no significant difference between the CPU bound and Not- CPU bound values, where the pattern looks similar in all the plots.

1.2 Impute null values if present, also check for the values which are equal to zero. Do they have any meaning or do we need to change them or drop them? Check for the possibility of creating new features if required. Also check for outliers and duplicates if there.

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

Figure 21: Null values

As shown in the above fig 16, there are 104 null values present in the rchar variable and 15 null values in the char variable, and those are imputed using the median function. The below images show the dataset after imputing the null values.

```
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
dtype: int64
```

Figure 23: Null Values after imputing

The below fig 18 shows the values that are equal to zeros. As we can see, there is some variable where 50% to 70% of the values are equal to zero. Imputing or dropping them will lead to a biased model. Therefore, the zeros will not be dropped from the dataset by accepting the zero could be a true value.

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

Figure 25: Zero values in the data

There is no duplicates in the dataset.

```
Number of duplicate rows = 0
```

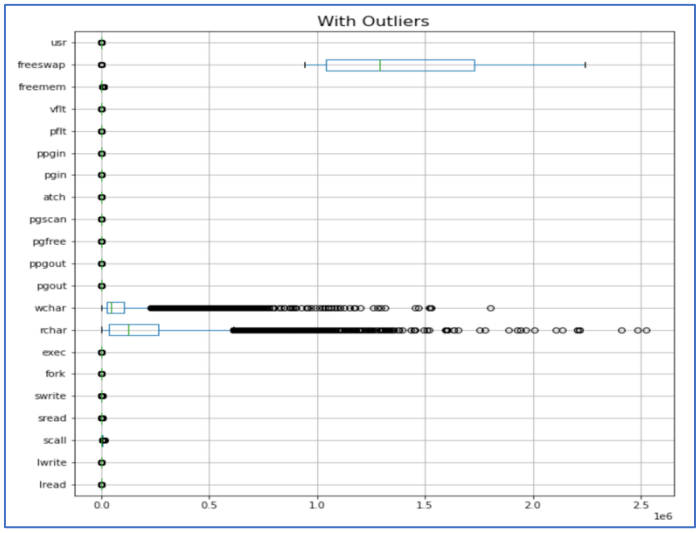


Figure 27: Dataset with Outliers

The above pic 19 show the dataset with outliers – surprisingly we could see outliers in all the variables. And it is important to treat the outliers since the linear regression is sensitive to outliers and could lead to a biased result. IQR (Interquartile Range) and boxplot methods used in identify the outliers. The target variables 'usr' have been ignored before analysing and treating the outliers because it could manipulate the data.

Outliers has been removed from all the independent variable in the below fig. 20

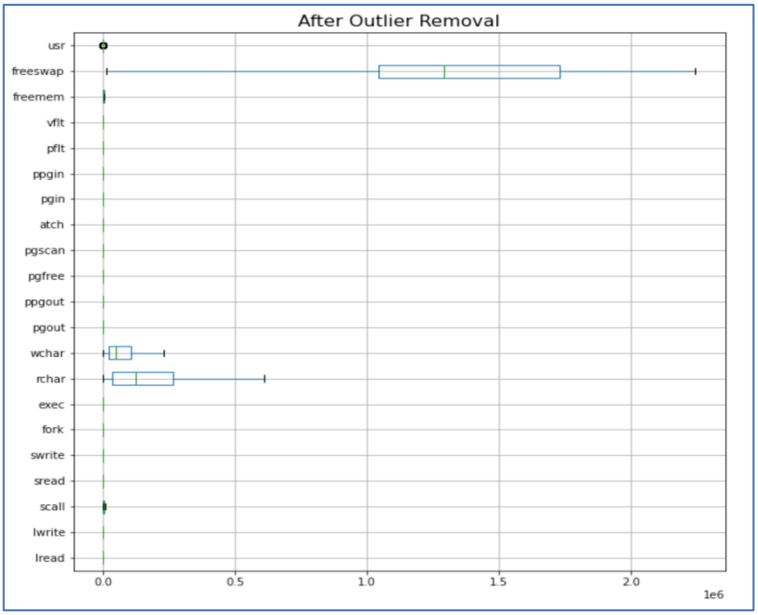


Figure 29: After Removal of outliers

The above Fig 20 Boxplot shows after treating the outliers. (By using IQR Method – a barrier has been created by taking the 1.5 x IQR and subtracting with the Q1, and adding with the Q3

1.3 Encode the data (having string values) for Modelling. Split the data into train and test (70:30). Apply Linear regression using scikit learn. Perform checks for significant variables using appropriate method from statsmodel. Create multiple models and check the performance of Predictions on Train and Test sets using Rsquare, RMSE & Adj Rsquare. Compare these models and select the best one with appropriate reasoning.

Encoding the string based values:

Dummies encoding will be for the only string based values which is 'runqsz' variable. One of two columns will be dropped to guarantee that the two columns are not linearly dependent on one another.

scall	sread	swrite	fork	exec	rchar	wchar	pgout	...	pgscan	atch	pgin	ppgin	pflt	vflt	freemem	freeswap	usr	runqsz_Not_CPU_Bound
2147.0	79.0	68.0	0.2	0.2	40671.0	53995.0	0.0	...	0.0	0.0	1.6	2.6	16.00	26.40	4659.125	1730946.0	95	0
170.0	18.0	21.0	0.2	0.2	448.0	8385.0	0.0	...	0.0	0.0	0.0	0.0	15.63	16.83	4659.125	1869002.0	97	1
2162.0	159.0	119.0	2.0	2.4	125473.5	31950.0	0.0	...	0.0	1.2	6.0	9.4	150.20	220.20	702.000	1021237.0	87	1
160.0	12.0	16.0	0.2	0.2	125473.5	8670.0	0.0	...	0.0	0.0	0.2	0.2	15.60	16.80	4659.125	1863704.0	98	1
330.0	39.0	38.0	0.4	0.4	125473.5	12185.0	0.0	...	0.0	0.0	1.0	1.2	37.80	47.60	633.000	1760253.0	90	1

Figure 31: Encoding the string based values

We will be separating the x and y since the independent feature will be only used to train the model; it will be used to predict the target variable.

X variable will be created by including only the independent variable, Therefore, target variable will be dropped –'usr'.

Y variable will be created by only keeping the target variable which is – 'usr'

First 5 rows of X & Y variable:

scall	sread	swrite	fork	exec	rchar	wchar	pgout	...	pgfree	pgscan	atch	pgin	ppgin	pflt	vflt	freemem	freeswap	runqsz_Not_CPU_Bound
47.0	79.0	68.0	0.2	0.2	40671.0	53995.0	0.0	...	0.0	0.0	0.0	1.6	2.6	16.00	26.40	4659.125	1730946.0	0
70.0	18.0	21.0	0.2	0.2	448.0	8385.0	0.0	...	0.0	0.0	0.0	0.0	0.0	15.63	16.83	4659.125	1869002.0	1
62.0	159.0	119.0	2.0	2.4	125473.5	31950.0	0.0	...	0.0	0.0	1.2	6.0	9.4	150.20	220.20	702.000	1021237.0	1
60.0	12.0	16.0	0.2	0.2	125473.5	8670.0	0.0	...	0.0	0.0	0.0	0.2	0.2	15.60	16.80	4659.125	1863704.0	1
30.0	39.0	38.0	0.4	0.4	125473.5	12185.0	0.0	...	0.0	0.0	0.0	1.0	1.2	37.80	47.60	633.000	1760253.0	1

```
0    95
1    97
2    87
3    98
4    90
Name: usr, dtype: int64
```

Figure 33:First 5 rows of X & Y variable

Train – Test Spilt

Data will spilt into train and test in 70:30 ratio. 70% of the data would be for training and 30% will be for testing using **sk.learn model**

Applying linear regression using sk.learn model.

Linear regression model has been imported and the training data has been fitted into the model.

The coefficient for the each independent attributes are below:

```
The coefficient for lread is -0.07504073422791643
The coefficient for lwrite is 0.03962723614635781
The coefficient for scall is 0.0010985026780728682
The coefficient for sread is 0.0009835373053474251
The coefficient for swrite is -0.00536019998511177
The coefficient for fork is -0.9026917050346785
The coefficient for exec is -0.05366220977150292
The coefficient for rchar is -1.0384453343370223e-05
The coefficient for wchar is -6.735870188054327e-06
The coefficient for pgout is -0.7157741777822665
The coefficient for ppgout is -0.01775655168781126
The coefficient for pgfree is 0.1048156143036473
The coefficient for pgscan is 0.0
The coefficient for atch is 1.0793198445095822
The coefficient for pgin is 0.29866551270177544
The coefficient for ppgin is -0.1703584737689542
The coefficient for pflt is -0.05480935758425474
The coefficient for vflt is 0.015459978615248702
The coefficient for freemem is -0.0023615312516139116
The coefficient for freeswap is 3.2657235608878066e-05
The coefficient for runqsz_Not_CPU_Bound is 7.01168416481982
```

Figure 35:coefficient for the each independent attributes

Intercept of the model is ***45.76001650301086***

R square:

R square is a metric that defines the variance ratio in the dependent variable that the independent variable can explain. Basically, How good the data fit in the regression model.

R square of training data:

The training data is passed into the score method that gives the R Square value, which is

```
0.6198709045647477
```

The Training data could capture only 61% of the variation in the 'usr' is explained by predictors in the model.

R square of testing data is `0.6110918220169601`

As we could performance of train data and test data is same. We can confirm the model is **stable.**

RMSE:
Root mean square error help us in understanding the difference between the estimate and actual value of the parameter of the model.

RMSE of Training data: ***11.155491739897576***
RMSE of Testing data : ***11.906167025535279***

Again, we can notice the performance of Test and train data is almost same.

Linear Regression using statsmodels:

Statsmodels is used to interrupt the linear regression model.

The X and y variables will be combined using the concat method for test and train data before initiating the stats model. And the dependent and independent features will be defined as an expression. Furthermore, the spaces in the column name will be replaced with an underscore (_) to put in the expressions to predict the 'usr'.

A model is build using lm1 from statsmodel by passing in the expression and training data.

Below fig 24 is the parameter for each independent attributes using statsmodels: It is clear that coefficient of each attributes in sklearn and statsmodels are the same.

```
Intercept                4.576002e+01
lread                   -7.504073e-02
lwrite                   3.962724e-02
scall                    1.098503e-03
sread                    9.835373e-04
swrite                  -5.360200e-03
fork                    -9.026917e-01
exec                    -5.366221e-02
rchar                   -1.038445e-05
wchar                   -6.735870e-06
pgout                   -7.157742e-01
ppgout                  -1.775655e-02
pgfree                   1.048156e-01
pgscan                  -1.096526e-14
atch                     1.079320e+00
pgin                     2.986655e-01
ppgin                   -1.703585e-01
pflt                    -5.480936e-02
vflt                     1.545998e-02
freemem                 -2.361531e-03
freeswap                 3.265724e-05
runqsz_Not_CPU_Bound     7.011684e+00
dtype: float64
```

Figure 37:parameter for each independent attributes

Summary of the model:

OLS Regression Results

Dep. Variable:	usr	R-squared:	0.620
Model:	OLS	Adj. R-squared:	0.619
Method:	Least Squares	F-statistic:	465.8
Date:	Wed, 01 Feb 2023	Prob (F-statistic):	0.00
Time:	13:33:49	Log-Likelihood:	-21966.
No. Observations:	5734	AIC:	4.397e+04
Df Residuals:	5713	BIC:	4.411e+04
Df Model:	20		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	45.7600	0.798	57.348	0.000	44.196	47.324
lread	-0.0750	0.023	-3.312	0.001	-0.119	-0.031
lwrite	0.0396	0.033	1.197	0.232	-0.025	0.105
scall	0.0011	0.000	6.927	0.000	0.001	0.001
sread	0.0010	0.003	0.386	0.699	-0.004	0.006
swrite	-0.0054	0.004	-1.480	0.139	-0.012	0.002
fork	-0.9027	0.333	-2.713	0.007	-1.555	-0.250
exec	-0.0537	0.130	-0.412	0.681	-0.309	0.202
rchar	-1.038e-05	1.23e-06	-8.439	0.000	-1.28e-05	-7.97e-06
wchar	-6.736e-06	2.61e-06	-2.584	0.010	-1.18e-05	-1.63e-06
pgout	-0.7158	0.227	-3.151	0.002	-1.161	-0.270
ppgout	-0.0178	0.199	-0.089	0.929	-0.407	0.372
pgfree	0.1048	0.121	0.870	0.384	-0.131	0.341
pgscan	-1.097e-14	3.41e-16	-32.201	0.000	-1.16e-14	-1.03e-14
atch	1.0793	0.361	2.994	0.003	0.373	1.786
pgin	0.2987	0.072	4.161	0.000	0.158	0.439
ppgin	-0.1704	0.050	-3.423	0.001	-0.268	-0.073
pflt	-0.0548	0.005	-10.958	0.000	-0.065	-0.045
vflt	0.0155	0.004	4.293	0.000	0.008	0.023
freemem	-0.0024	0.000	-18.443	0.000	-0.003	-0.002
freeswap	3.266e-05	4.8e-07	68.079	0.000	3.17e-05	3.36e-05
runqsz_Not_CPU_Bound	7.0117	0.318	22.046	0.000	6.388	7.635

Omnibus:	1424.388	Durbin-Watson:	2.061
Prob(Omnibus):	0.000	Jarque-Bera (JB):	4185.515
Skew:	-1.287	Prob(JB):	0.00
Kurtosis:	6.301	Cond. No.	8.18e+22

Figure 41: Initial model summary

VIF values:

lread	9.505097
lwrite	6.548895
scall	8.826652
sread	18.181101
swrite	16.779288
fork	25.002626
exec	6.092824
rchar	4.345751
wchar	3.341629
pgout	16.004553
ppgout	40.798798
pgfree	22.973384
pgscan	NaN
atch	2.735390
pgin	23.072154
ppgin	23.163752
pflt	24.685414
vflt	33.875742
freemem	3.406296
freeswap	7.112881
runqsz_Not_CPU_Bound	2.155770

dtype: float64

Figure 39: VIF Values

R-squared is : 0.620
Adj. R-squared: 0.619

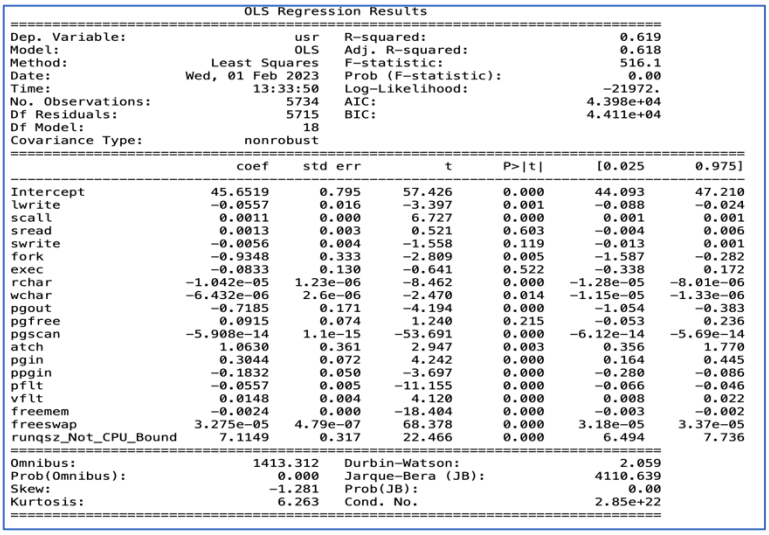
As we could see in the summary, there are few features that has a higher p-value, which has to be dropped one by one. Before dropping them, multicollinearity should be checked. Since the Coefficient can't be trusted as representing its relationship with the target variable –'usr', because there high chance where other variables could have relationship with the specific variable.

VIF (Variance inflation factor) measure is used to check the multicollinearity in all the independent variables (x variable) – to check if one x variable is predicated by another x variable. If the VIF value is 1, then there is no correlation between the predictors and the remaining predictors. If the VIF value exceeds 5, then there is moderate multicollinearity; if the VIF value is ten and above, it shows a high sign of multicollinearity.

The VIF value of ppgout, vflt, pflt, ppgin, pgin, pgfree, pgout, fork, swrite, sread, scall, lwrite and lread are correlated with one or more independent variables. We will drop one by one all variable that has a VIF value of more than 5 to treat multicollinearity. Adjusted R-squared of the model should have only the least impact while dropping the features.

Let's drop the variable that has highest VIF value to observe the effect on the predictive model:

Summary after dropping the ppgout: (Number of pages, paged out per second)

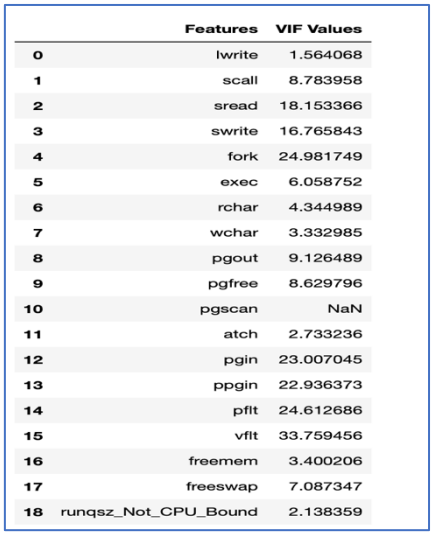


OLS Regression Results

Dep. Variable:	usr	R-squared:	0.619
Model:	OLS	Adj. R-squared:	0.618
Method:	Least Squares	F-statistic:	516.1
Date:	Wed, 01 Feb 2023	Prob (F-statistic):	0.00
Time:	13:33:50	Log-Likelihood:	-21972.
No. Observations:	5734	AIC:	4.398e+04
Df Residuals:	5715	BIC:	4.411e+04
Df Model:	18		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	45.6519	0.795	57.426	0.000	44.093	47.210
lwrite	-0.0557	0.016	-3.397	0.001	-0.088	-0.024
scall	0.0011	0.000	6.727	0.000	0.001	0.001
sread	0.0013	0.003	0.521	0.603	-0.004	0.006
swrite	-0.0056	0.004	-1.558	0.119	-0.013	0.001
fork	-0.9348	0.333	-2.809	0.005	-1.587	-0.282
exec	-0.0833	0.130	-0.641	0.522	-0.338	0.172
rchar	-1.042e-05	1.23e-06	-8.462	0.000	-1.28e-05	-8.01e-06
wchar	-6.432e-06	2.6e-06	-2.470	0.014	-1.15e-05	-1.33e-06
pgout	-0.7185	0.171	-4.194	0.000	-1.054	-0.383
pgfree	0.0915	0.074	1.240	0.215	-0.053	0.236
pgscan	-5.908e-14	1.1e-15	-53.691	0.000	-6.12e-14	-5.69e-14
atch	1.0630	0.361	2.947	0.003	0.356	1.770
pgin	0.3044	0.072	4.242	0.000	0.164	0.445
ppgin	-0.1832	0.050	-3.697	0.000	-0.280	-0.086
pflt	-0.0557	0.005	-11.155	0.000	-0.066	-0.046
vflt	0.0148	0.004	4.120	0.000	0.008	0.022
freemem	-0.0024	0.000	-18.404	0.000	-0.003	-0.002
freeswap	3.275e-05	4.79e-07	68.378	0.000	3.18e-05	3.37e-05
runqsz_Not_CPU_Bound	7.1149	0.317	22.466	0.000	6.494	7.736

Omnibus:	1413.312	Durbin-Watson:	2.059
Prob(Omnibus):	0.000	Jarque-Bera (JB):	4110.639
Skew:	-1.281	Prob(JB):	0.00
Kurtosis:	6.263	Cond. No.	2.85e+22



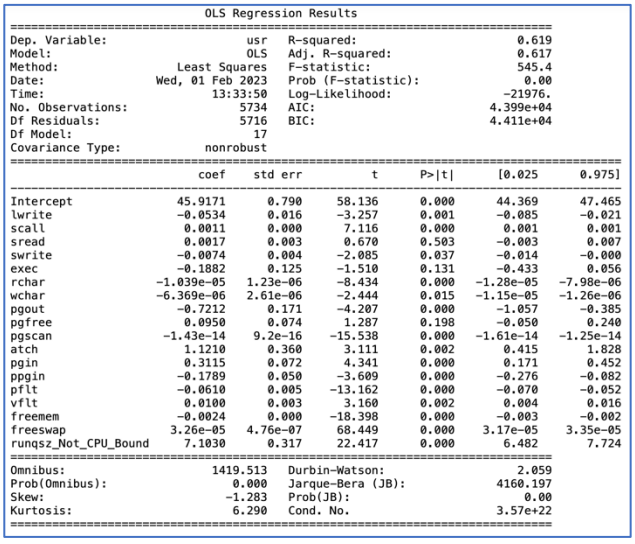
	Features	VIF Values
0	lwrite	1.564068
1	scall	8.783958
2	sread	18.153366
3	swrite	16.765843
4	fork	24.981749
5	exec	6.058752
6	rchar	4.344989
7	wchar	3.332985
8	pgout	9.126489
9	pgfree	8.629796
10	pgscan	NaN
11	atch	2.733236
12	pgin	23.007045
13	ppgin	22.936373
14	pflt	24.612686
15	vflt	33.759456
16	freemem	3.400206
17	freeswap	7.087347
18	runqsz_Not_CPU_Bound	2.138359

Figure 43:Summary after dropping the ppgout

R – squared: 0.619; Adj. R squared: 0.618

As we see in the above summary, on dropping the ppgout, the R square and Adj R-square decreased by 0.001. It is clear that there is no significant difference in VIF value. So we continue further dropping the variable that has the highest VIF score.

Summary after dropping the fork variable: (Number of system fork calls per second.)

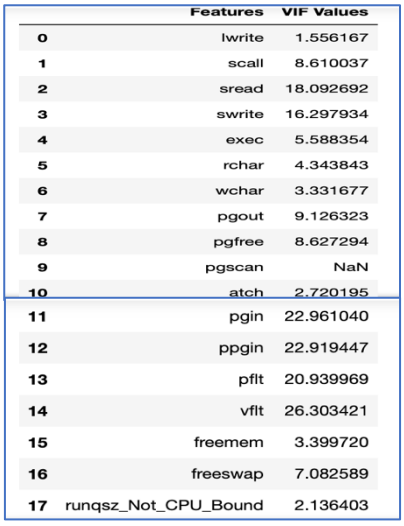


OLS Regression Results

Dep. Variable:	usr	R-squared:	0.619
Model:	OLS	Adj. R-squared:	0.617
Method:	Least Squares	F-statistic:	545.4
Date:	Wed, 01 Feb 2023	Prob (F-statistic):	0.00
Time:	13:33:50	Log-Likelihood:	-21976.
No. Observations:	5734	AIC:	4.399e+04
Df Residuals:	5716	BIC:	4.411e+04
Df Model:	17		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	45.9171	0.790	58.136	0.000	44.369	47.465
lwrite	-0.0534	0.016	-3.257	0.001	-0.085	-0.021
scall	0.0011	0.000	7.116	0.000	0.001	0.001
sread	0.0017	0.003	0.670	0.503	-0.003	0.007
swrite	-0.0074	0.004	-2.085	0.037	-0.014	-0.000
exec	-0.1882	0.125	-1.510	0.131	-0.433	0.056
rchar	-1.039e-05	1.23e-06	-8.434	0.000	-1.28e-05	-7.98e-06
wchar	-6.369e-06	2.61e-06	-2.444	0.015	-1.15e-05	-1.26e-06
pgout	-0.7212	0.171	-4.207	0.000	-1.057	-0.385
pgfree	0.0950	0.074	1.287	0.198	-0.050	0.240
pgscan	-1.43e-14	9.2e-16	-15.538	0.000	-1.61e-14	-1.25e-14
atch	1.1210	0.360	3.111	0.002	0.415	1.828
pgin	0.3115	0.072	4.341	0.000	0.171	0.452
ppgin	-0.1789	0.050	-3.609	0.000	-0.276	-0.082
pflt	-0.0610	0.005	-13.162	0.000	-0.070	-0.052
vflt	0.0100	0.003	3.160	0.002	0.004	0.016
freemem	-0.0024	0.000	-18.398	0.000	-0.003	-0.002
freeswap	3.26e-05	4.76e-07	68.449	0.000	3.17e-05	3.35e-05
runqsz_Not_CPU_Bound	7.1030	0.317	22.417	0.000	6.482	7.724

Omnibus:	1419.513	Durbin-Watson:	2.059
Prob(Omnibus):	0.000	Jarque-Bera (JB):	4160.197
Skew:	-1.283	Prob(JB):	0.00
Kurtosis:	6.290	Cond. No.	3.57e+22

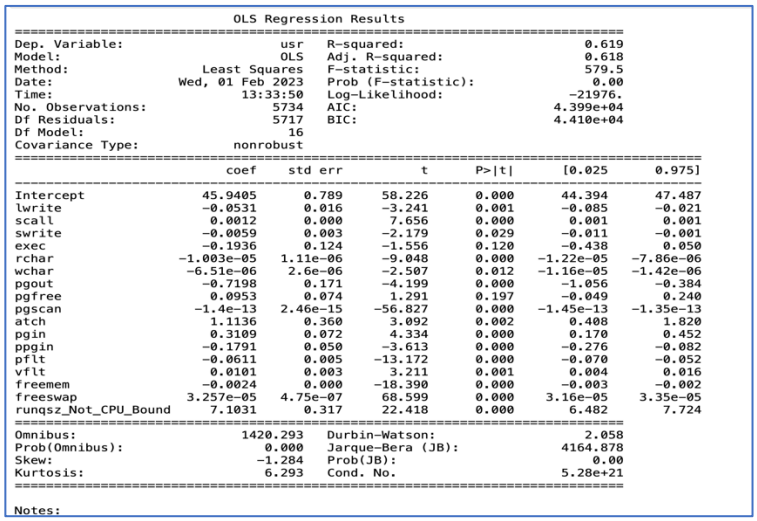


	Features	VIF Values
0	lwrite	1.556167
1	scall	8.610037
2	sread	18.092692
3	swrite	16.297934
4	exec	5.588354
5	rchar	4.343843
6	wchar	3.331677
7	pgout	9.126323
8	pgfree	8.627294
9	pgscan	NaN
10	atch	2.720195
11	pgin	22.961040
12	ppgin	22.919447
13	pflt	20.939969
14	vflt	26.303421
15	freemem	3.399720
16	freeswap	7.082589
17	runqsz_Not_CPU_Bound	2.136403

Figure 45:Summary after dropping the fork variable:

R – squared: 0.619; Adj. R squared: 0.617
As we could see from the above summary, after dropping the fork variable. There is only minor effect on adjusted R square (0.001), there for we can remove it from the training dataset. Furthermore, we can still see some of the variable have high multicollinearity.

Dropping the sread variable:



OLS Regression Results

Dep. Variable:	usr	R-squared:	0.619
Model:	OLS	Adj. R-squared:	0.618
Method:	Least Squares	F-statistic:	579.5
Date:	Wed, 01 Feb 2023	Prob (F-statistic):	0.00
Time:	13:33:50	Log-Likelihood:	-21976.
No. Observations:	5734	AIC:	4.399e+04
Df Residuals:	5717	BIC:	4.410e+04
Df Model:	16		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	45.9405	0.789	58.226	0.000	44.394	47.487
lwrite	-0.0531	0.016	-3.241	0.001	-0.085	-0.021
scall	0.0012	0.000	7.656	0.000	0.001	0.001
swrite	-0.0059	0.003	-2.179	0.029	-0.011	-0.001
exec	-0.1936	0.124	-1.556	0.120	-0.438	0.050
rchar	-1.003e-05	1.11e-06	-9.048	0.000	-1.22e-05	-7.86e-06
wchar	-6.51e-06	2.6e-06	-2.507	0.012	-1.16e-05	-1.42e-06
pgout	-0.7198	0.171	-4.199	0.000	-1.056	-0.384
pgfree	0.0953	0.074	1.291	0.197	-0.049	0.240
pgscan	-1.4e-13	2.46e-15	-56.827	0.000	-1.45e-13	-1.35e-13
atch	1.1136	0.360	3.092	0.002	0.408	1.820
pgin	0.3109	0.072	4.334	0.000	0.170	0.452
ppgin	-0.1791	0.050	-3.613	0.000	-0.276	-0.082
pflt	-0.0611	0.005	-13.172	0.000	-0.070	-0.052
vflt	0.0101	0.003	3.211	0.001	0.004	0.016
freemem	-0.0024	0.000	-18.390	0.000	-0.003	-0.002
freeswap	3.257e-05	4.75e-07	68.599	0.000	3.16e-05	3.35e-05
runqsz_Not_CPU_Bound	7.1031	0.317	22.418	0.000	6.482	7.724

Omnibus:	1420.293	Durbin-Watson:	2.058
Prob(Omnibus):	0.000	Jarque-Bera (JB):	4164.878
Skew:	-1.284	Prob(JB):	0.00
Kurtosis:	6.293	Cond. No.	5.28e+21

Notes:

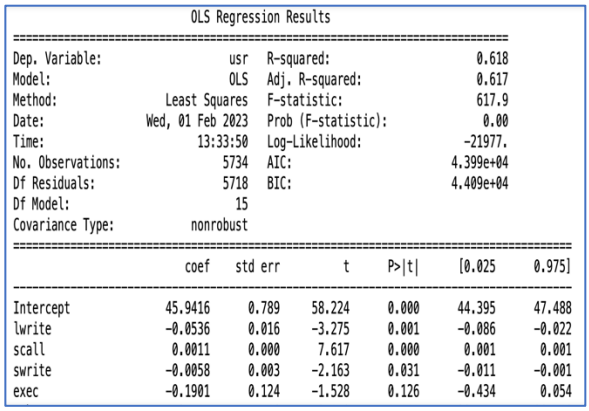
	Features	VIF Values
0	lwrite	1.554461
1	scall	7.799333
2	swrite	9.184514
3	exec	5.566845
4	rchar	3.500320
5	wchar	3.311998
6	pgout	9.124731
7	pgfree	8.626990
8	pgscan	NaN
9	atch	2.718312
10	pgin	22.959654
11	ppgin	22.918164
12	pflt	20.933145
13	vflt	26.181175
14	freemem	3.399255
15	freeswap	7.050035
16	runqsz_Not_CPU_Bound	2.135769

Figure 47:Summary after Dropping the sread variable

R – squared: 0.619; Adj. R squared: 0.618

As we could see from the above summary, after dropping the sread variable. Adj. R-square is increased by 0.001. Therefore we can remove it from the training dataset. Furthermore, we can still see some of the variable have high multicollinearity, so we can continue dropping the variable that has higher multicollinearity.

Dropping the pgfree variable:



OLS Regression Results

Dep. Variable:	usr	R-squared:	0.618
Model:	OLS	Adj. R-squared:	0.617
Method:	Least Squares	F-statistic:	617.9
Date:	Wed, 01 Feb 2023	Prob (F-statistic):	0.00
Time:	13:33:50	Log-Likelihood:	-21977.
No. Observations:	5734	AIC:	4.399e+04
Df Residuals:	5718	BIC:	4.409e+04
Df Model:	15		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	45.9416	0.789	58.224	0.000	44.395	47.488
lwrite	-0.0536	0.016	-3.275	0.001	-0.086	-0.022
scall	0.0011	0.000	7.617	0.000	0.001	0.001
swrite	-0.0058	0.003	-2.163	0.031	-0.011	-0.001
exec	-0.1901	0.124	-1.528	0.126	-0.434	0.054

	Features	VIF Values
0	lwrite	1.553437
1	scall	7.790704
2	swrite	9.183108
3	exec	5.564156
4	rchar	3.500317
5	wchar	3.311849
6	pgout	2.899919
7	pgscan	NaN
8	atch	2.716974
9	pgin	22.899568
10	ppgin	22.701019
11	pflt	20.881441
12	vflt	26.076564
13	freemem	3.378971
14	freeswap	7.029390
15	runqsz_Not_CPU_Bound	2.134968

Figure 49: summary after Dropping the pgfree variable:

R – squared: 0.618; Adj. R squared: 0.617

As we could see from the above summary, after dropping the pgfree variable. Adj. R-square is only dropped by 0.001. Therefore we can remove it from the training dataset. Furthermore, we can still see some of the variable have high multicollinearity, so we can continue dropping the variable that has higher multicollinearity as we could see in the above fig. 30

Dropping the pgin variable:

OLS Regression Results

Dep. Variable:	usr	R-squared:	0.617
Model:	OLS	Adj. R-squared:	0.616
Method:	Least Squares	F-statistic:	658.8
Date:	Wed, 01 Feb 2023	Prob (F-statistic):	0.00
Time:	13:33:51	Log-Likelihood:	-21986.
No. Observations:	5734	AIC:	4.400e+04
Df Residuals:	5719	BIC:	4.410e+04
Df Model:	14		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	46.2153	0.788	58.676	0.000	44.671	47.759
lwrite	-0.0557	0.016	-3.401	0.001	-0.088	-0.024
scall	0.0012	0.000	7.720	0.000	0.001	0.001
swrite	-0.0059	0.003	-2.191	0.028	-0.011	-0.001
exec	-0.1811	0.125	-1.454	0.146	-0.425	0.063
rchar	-1.045e-05	1.11e-06	-9.445	0.000	-1.26e-05	-8.28e-06
wchar	-6.236e-06	2.6e-06	-2.398	0.016	-1.13e-05	-1.14e-06
pgout	-0.5406	0.097	-5.586	0.000	-0.730	-0.351
pgscan	6.031e-14	1.21e-15	49.975	0.000	5.79e-14	6.27e-14
atch	1.1326	0.361	3.140	0.002	0.426	1.840
ppgin	0.0254	0.017	1.506	0.132	-0.008	0.058
pflt	-0.0630	0.005	-13.630	0.000	-0.072	-0.054
vflt	0.0118	0.003	3.772	0.000	0.006	0.018
freemem	-0.0024	0.000	-18.501	0.000	-0.003	-0.002
freeswap	3.245e-05	4.74e-07	68.434	0.000	3.15e-05	3.34e-05
runqsz_Not_CPU_Bound	7.0874	0.317	22.338	0.000	6.465	7.709

Omnibus:	1433.148	Durbin-Watson:	2.055
Prob(Omnibus):	0.000	Jarque-Bera (JB):	4225.990
Skew:	-1.293	Prob(JB):	0.00
Kurtosis:	6.316	Cond. No.	8.71e+20

	Features	VIF Values
0	lwrite	1.552988
1	scall	7.777049
2	swrite	9.182952
3	exec	5.561383
4	rchar	3.477722
5	wchar	3.307949
6	pgout	2.899776
7	pgscan	NaN
8	atch	2.716453
9	ppgin	2.596836
10	pflt	20.699718
11	vflt	25.739354
12	freemem	3.378925
13	freeswap	7.029145
14	runqsz_Not_CPU_Bound	2.133733

Figure 51: Summary after dropping the pgin variable

R – squared: 0.617; Adj. R squared: 0.616

As we could see from the above summary, after dropping the pgin variable. Adj. R-square is only dropped by 0.001. Therefore we can remove it from the training dataset. Furthermore, we can still see some of the variable have high multicollinearity, so we can continue dropping the variable that has higher multicollinearity as we could see in the above fig. 31

Dropping the vflt variable:

OLS Regression Results

Dep. Variable:	usr	R-squared:	0.616
Model:	OLS	Adj. R-squared:	0.615
Method:	Least Squares	F-statistic:	706.7
Date:	Wed, 01 Feb 2023	Prob (F-statistic):	0.00
Time:	13:33:51	Log-Likelihood:	-21993.
No. Observations:	5734	AIC:	4.401e+04
Df Residuals:	5720	BIC:	4.411e+04
Df Model:	13		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	46.4587	0.786	59.115	0.000	44.918	47.999
lwrite	-0.0534	0.016	-3.259	0.001	-0.086	-0.021
scall	0.0012	0.000	7.772	0.000	0.001	0.001
swrite	-0.0042	0.003	-1.586	0.113	-0.009	0.001
exec	-0.0826	0.122	-0.678	0.498	-0.322	0.156
rchar	-1.001e-05	1.1e-06	-9.089	0.000	-1.22e-05	-7.85e-06
wchar	-7.669e-06	2.58e-06	-2.978	0.003	-1.27e-05	-2.62e-06
pgout	-0.5220	0.097	-5.394	0.000	-0.712	-0.332
pgscan	1.07e-13	1.92e-15	55.612	0.000	1.03e-13	1.11e-13
atch	1.2127	0.360	3.365	0.001	0.506	1.919
ppgin	0.0429	0.016	2.648	0.008	0.011	0.075
pflt	-0.0486	0.003	-18.581	0.000	-0.054	-0.043
freemem	-0.0024	0.000	-18.387	0.000	-0.003	-0.002
freeswap	3.219e-05	4.7e-07	68.536	0.000	3.13e-05	3.31e-05
runqsz_Not_CPU_Bound	7.0966	0.318	22.342	0.000	6.474	7.719

Omnibus:	1468.706	Durbin-Watson:	2.056
Prob(Omnibus):	0.000	Jarque-Bera (JB):	4376.117
Skew:	-1.322	Prob(JB):	0.00
Kurtosis:	6.366	Cond. No.	5.17e+21

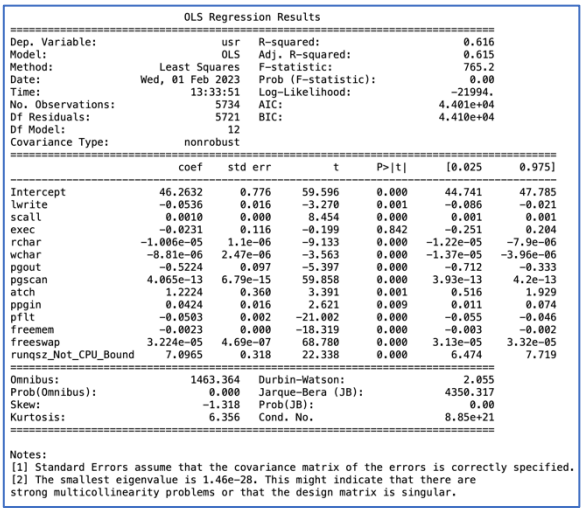
	Features	VIF Values
0	lwrite	1.548848
1	scall	7.769240
2	swrite	8.886709
3	exec	5.301284
4	rchar	3.433221
5	wchar	3.243546
6	pgout	2.891588
7	pgscan	NaN
8	atch	2.704005
9	ppgin	2.379016
10	pflt	6.651752
11	freemem	3.377637
12	freeswap	6.901227
13	runqsz_Not_CPU_Bound	2.130458

Figure 53: Summary after dropping vflt variable

R – squared: 0.616; Adj. R squared: 0.615

As we could see from the above summary, after dropping the vflt variable. R square & Adj. R-square is only dropped by 0.001. Therefore we can remove it from the training dataset. Furthermore, we can still see some of the variable have high multicollinearity, so we can continue dropping the variable that has higher multicollinearity as we could see in the above fig. 32

Dropping the swrite variable:



OLS Regression Results

Dep. Variable:	usr	R-squared:	0.616
Model:	OLS	Adj. R-squared:	0.615
Method:	Least Squares	F-statistic:	765.2
Date:	Wed, 01 Feb 2023	Prob (F-statistic):	0.00
Time:	13:33:51	Log-Likelihood:	-21994.
No. Observations:	5734	AIC:	4.401e+04
Df Residuals:	5721	BIC:	4.410e+04
Df Model:	12		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	46.2632	0.776	59.596	0.000	44.741	47.785
lwrite	-0.0536	0.016	-3.270	0.001	-0.086	-0.021
scall	0.0010	0.000	8.454	0.000	0.001	0.001
exec	-0.0231	0.116	-0.199	0.842	-0.251	0.204
rchar	-1.006e-05	1.1e-06	-9.133	0.000	-1.22e-05	-7.9e-06
wchar	-8.81e-06	2.47e-06	-3.563	0.000	-1.37e-05	-3.96e-06
pgout	-0.5224	0.097	-5.397	0.000	-0.712	-0.333
pgscan	4.065e-13	6.79e-15	59.858	0.000	3.93e-13	4.2e-13
atch	1.2224	0.360	3.391	0.001	0.516	1.929
ppgin	0.0424	0.016	2.621	0.009	0.011	0.074
pflt	-0.0503	0.002	-21.002	0.000	-0.055	-0.046
freemem	-0.0023	0.000	-18.319	0.000	-0.003	-0.002
freeswap	3.224e-05	4.69e-07	68.780	0.000	3.13e-05	3.32e-05
runqsz_Not_CPU_Bound	7.0965	0.318	22.338	0.000	6.474	7.719

Omnibus:	1463.364	Durbin-Watson:	2.055
Prob(Omnibus):	0.000	Jarque-Bera (JB):	4350.317
Skew:	-1.318	Prob(JB):	0.00
Kurtosis:	6.356	Cond. No.	8.85e+21

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The smallest eigenvalue is 1.46e-28. This might indicate that there are
strong multicollinearity problems or that the design matrix is singular.

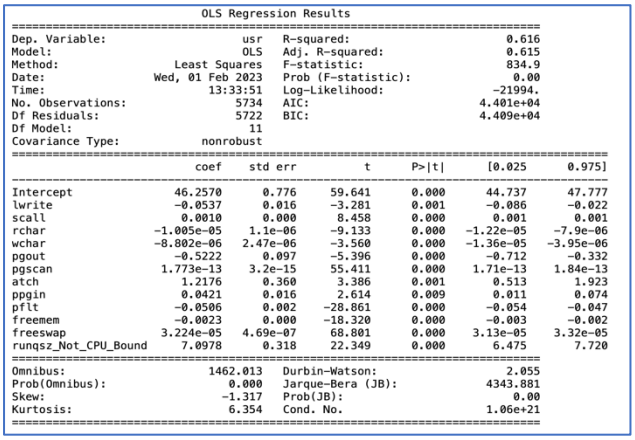
	Features	VIF Values
0	lwrite	1.547027
1	scall	4.624351
2	exec	4.827719
3	rchar	3.426993
4	wchar	2.957566
5	pgout	2.891443
6	pgscan	NaN
7	atch	2.704005
8	ppgin	2.374736
9	pflt	5.620649
10	freemem	3.330078
11	freeswap	6.826774
12	runqsz_Not_CPU_Bound	2.122703

Figure 55: Summary dropping the swrite variable:

R – squared: 0.616; Adj. R squared: 0.615

As we could see from the above summary, after dropping the swrite variable. R square & Adj. R-square remain the same. Therefore we can remove it from the training dataset. Furthermore, we can still see pflt & freeswap have high multicollinearity, so we can continue dropping the variable that has higher multicollinearity as we could see in the above fig. 33

Dropping the exec variable:



OLS Regression Results

Dep. Variable:	usr	R-squared:	0.616
Model:	OLS	Adj. R-squared:	0.615
Method:	Least Squares	F-statistic:	834.9
Date:	Wed, 01 Feb 2023	Prob (F-statistic):	0.00
Time:	13:33:51	Log-Likelihood:	-21994.
No. Observations:	5734	AIC:	4.401e+04
Df Residuals:	5722	BIC:	4.409e+04
Df Model:	11		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	46.2570	0.776	59.641	0.000	44.737	47.777
lwrite	-0.0537	0.016	-3.281	0.001	-0.086	-0.022
scall	0.0010	0.000	8.458	0.000	0.001	0.001
rchar	-1.005e-05	1.1e-06	-9.133	0.000	-1.22e-05	-7.9e-06
wchar	-8.802e-06	2.47e-06	-3.560	0.000	-1.36e-05	-3.95e-06
pgout	-0.5222	0.097	-5.396	0.000	-0.712	-0.332
pgscan	1.773e-13	3.2e-15	55.411	0.000	1.71e-13	1.84e-13
atch	1.2176	0.360	3.386	0.001	0.513	1.923
ppgin	0.0421	0.016	2.614	0.009	0.011	0.074
pflt	-0.0506	0.002	-28.861	0.000	-0.054	-0.047
freemem	-0.0023	0.000	-18.320	0.000	-0.003	-0.002
freeswap	3.224e-05	4.69e-07	68.801	0.000	3.13e-05	3.32e-05
runqsz_Not_CPU_Bound	7.0978	0.318	22.349	0.000	6.475	7.720

Omnibus:	1462.013	Durbin-Watson:	2.055
Prob(Omnibus):	0.000	Jarque-Bera (JB):	4343.881
Skew:	-1.317	Prob(JB):	0.00
Kurtosis:	6.354	Cond. No.	1.06e+21

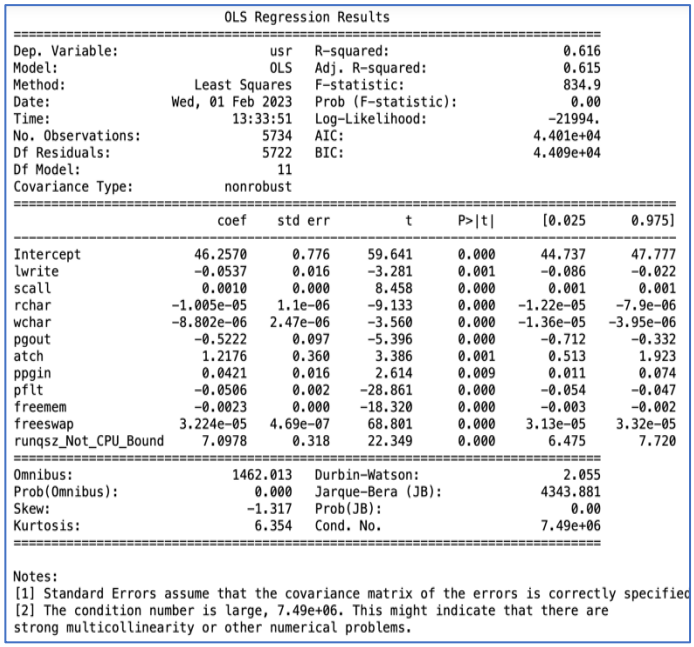
0	lwrite	1.543487
1	scall	4.593703
2	rchar	3.426992
3	wchar	2.957139
4	pgout	2.891116
5	pgscan	NaN
6	atch	2.690199
7	ppgin	2.346590
8	pflt	3.019915
9	freemem	3.327487
10	freeswap	6.822916
11	runqsz_Not_CPU_Bound	2.122601

Figure 57: Summary after Dropping the exec variable

R – squared: 0.616; Adj. R squared: 0.615

As we could see from the above summary, after dropping the exec variable. R square & Adj. R-square remain the same. Therefore we can remove it from the training dataset. Furthermore, we can still see freeswap has high multicollinearity, so we can continue dropping the variable that has higher multicollinearity as we could see in the above fig. 34.

Dropping the pgscan variable:



OLS Regression Results

Dep. Variable:	usr	R-squared:	0.616
Model:	OLS	Adj. R-squared:	0.615
Method:	Least Squares	F-statistic:	834.9
Date:	Wed, 01 Feb 2023	Prob (F-statistic):	0.00
Time:	13:33:51	Log-Likelihood:	-21994.
No. Observations:	5734	AIC:	4.401e+04
Df Residuals:	5722	BIC:	4.409e+04
Df Model:	11		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	46.2570	0.776	59.641	0.000	44.737	47.777
lwrite	-0.0537	0.016	-3.281	0.001	-0.086	-0.022
scall	0.0010	0.000	8.458	0.000	0.001	0.001
rchar	-1.005e-05	1.1e-06	-9.133	0.000	-1.22e-05	-7.9e-06
wchar	-8.802e-06	2.47e-06	-3.560	0.000	-1.36e-05	-3.95e-06
pgout	-0.5222	0.097	-5.396	0.000	-0.712	-0.332
atch	1.2176	0.360	3.386	0.001	0.513	1.923
ppgin	0.0421	0.016	2.614	0.009	0.011	0.074
pflt	-0.0506	0.002	-28.861	0.000	-0.054	-0.047
freemem	-0.0023	0.000	-18.320	0.000	-0.003	-0.002
freeswap	3.224e-05	4.69e-07	68.801	0.000	3.13e-05	3.32e-05
runqsz_Not_CPU_Bound	7.0978	0.318	22.349	0.000	6.475	7.720

Omnibus:	1462.013	Durbin-Watson:	2.055
Prob(Omnibus):	0.000	Jarque-Bera (JB):	4343.881
Skew:	-1.317	Prob(JB):	0.00
Kurtosis:	6.354	Cond. No.	7.49e+06

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified
[2] The condition number is large, 7.49e+06. This might indicate that there are
strong multicollinearity or other numerical problems.

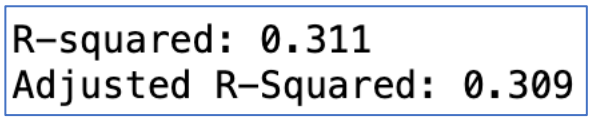
	Features	VIF Values
0	lwrite	1.543487
1	scall	4.593703
2	rchar	3.426992
3	wchar	2.957139
4	pgout	2.891116
5	atch	2.690199
6	ppgin	2.346590
7	pflt	3.019915
8	freemem	3.327487
9	freeswap	6.822916
10	runqsz_Not_CPU_Bound	2.122601

Figure 59: Summary after Dropping the pgscan variable

R – squared: 0.616; Adj. R squared: 0.615

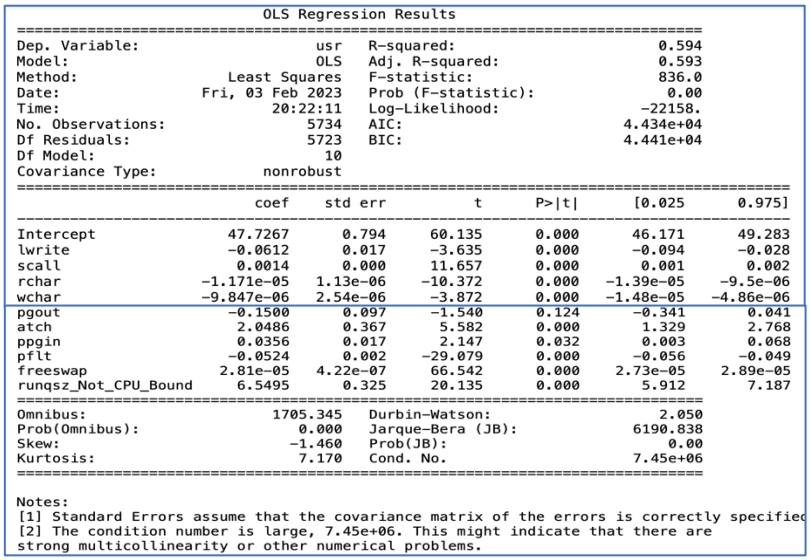
As we could see from the above summary, after dropping the pgscan variable. R square, Adj. R-square & VIF value remains the same.

As we could see the fig.59 freeswap is only variable that has high multicollinearity. When we tried dropping it. R square & adj. significantly dropped down to 0.311 to 0.309 from 0.616 & 0.615 respectively. Therefore the freeswap can't be dropped from the regression.



Dropping the freemem variable:

As we could see freeswap only variable that has high multicollinearity in the VIF table. When we tried dropping the freemem the high multicollinearity of Freeswap variable dropped from 6.82 to 3.66. Therefore, will drop the freemem from training set. When we check for the multicollinearity, It is evident that for the variable the VIF score is below 5. Now that we don't have any multicollinearity in our dataset. (refer fig. 61)



```
                            OLS Regression Results
==============================================================================
Dep. Variable:                    usr   R-squared:                       0.594
Model:                            OLS   Adj. R-squared:                  0.593
Method:                 Least Squares   F-statistic:                     836.0
Date:                Fri, 03 Feb 2023   Prob (F-statistic):               0.00
Time:                        20:22:11   Log-Likelihood:                -22158.
No. Observations:                5734   AIC:                         4.434e+04
Df Residuals:                    5723   BIC:                         4.441e+04
Df Model:                          10
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept     47.7267      0.794     60.135      0.000      46.171      49.283
lwrite        -0.0612      0.017     -3.635      0.000      -0.094      -0.028
scall          0.0014      0.000     11.657      0.000       0.001       0.002
rchar      -1.171e-05   1.13e-06    -10.372      0.000   -1.39e-05   -9.5e-06
wchar      -9.847e-06   2.54e-06     -3.872      0.000   -1.48e-05   -4.86e-06
pgout         -0.1500      0.097     -1.540      0.124      -0.341       0.041
atch           2.0486      0.367      5.582      0.000       1.329       2.768
ppgin          0.0356      0.017      2.147      0.032       0.003       0.068
pflt          -0.0524      0.002    -29.079      0.000      -0.056      -0.049
freeswap   2.81e-05   4.22e-07     66.542      0.000    2.73e-05    2.89e-05
runqsz_Not_CPU_Bound     6.5495      0.325     20.135      0.000       5.912       7.187
==============================================================================
Omnibus:                     1705.345   Durbin-Watson:                   2.050
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             6190.838
Skew:                          -1.460   Prob(JB):                         0.00
Kurtosis:                       7.170   Cond. No.                     7.45e+06
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specifie
[2] The condition number is large, 7.45e+06. This might indicate that there are
strong multicollinearity or other numerical problems.
```

	Features	VIF Values
0	lwrite	1.543410
1	scall	4.340468
2	rchar	3.409208
3	wchar	2.956873
4	pgout	2.758760
5	atch	2.638406
6	ppgin	2.346524
7	pflt	3.009636
8	freeswap	3.666801
9	runqsz_Not_CPU_Bound	2.114543

Figure 61: Summary after Dropping the freemem variable

However, the R square and adjusted R square dropped 0.22. Though, it is not a significant drop. This model has no multicollinearity issue, and moreover, the p-value of the coefficient has become reliable, and we can remove the non-significant predictor variable.

Based on the above summary, we are dropping ppgin, and pgout variable, which has high p-value, and there is no evidence that usr is affected by ppgin and pgout variables. From the below summary, it is clear that without the ppgin and pgout –The adjusted R square hasn't dropped. In a nutshell, after dropping the features causing strong multicollinearity and statistically insignificant features. And the model performance hasn't dropped sharply (0.594 to 0.593). This shows that these features don't have strong predictive power.

Final Model:

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                    usr   R-squared:                       0.593
Model:                            OLS   Adj. R-squared:                  0.593
Method:                 Least Squares   F-statistic:                     1044.
Date:                Fri, 03 Feb 2023   Prob (F-statistic):               0.00
Time:                        21:44:05   Log-Likelihood:                -22161.
No. Observations:                5734   AIC:                         4.434e+04
Df Residuals:                    5725   BIC:                         4.440e+04
Df Model:                           8
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept     47.9263      0.780     61.473      0.000      46.398      49.455
lwrite        -0.0596      0.017     -3.537      0.000      -0.093      -0.027
scall          0.0014      0.000     11.742      0.000       0.001       0.002
rchar      -1.121e-05    1.1e-06    -10.147      0.000   -1.34e-05   -9.04e-06
wchar      -9.934e-06   2.53e-06     -3.923      0.000   -1.49e-05   -4.97e-06
atch           1.8333      0.303      6.050      0.000       1.239       2.427
pflt          -0.0523      0.002    -29.113      0.000      -0.056      -0.049
freeswap   2.801e-05   4.13e-07     67.837      0.000    2.72e-05    2.88e-05
runqsz_Not_CPU_Bound     6.5335      0.325     20.131      0.000       5.897       7.170
==============================================================================
Omnibus:                     1723.692   Durbin-Watson:                   2.052
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             6312.622
Skew:                          -1.473   Prob(JB):                         0.00
Kurtosis:                       7.212   Cond. No.                     7.34e+06
==============================================================================
```

Figure 63: Final Linear Regression Model

The final model is free from high multicollinearity and high p-value(>0.05). However, R-squared revels the regression model only fully accounts for 59% of the variability in the target variable. And the above model shows that these variable did not have much predictive power.

Assumption of linear regression:

Our first assumption will be the Linearity and the Independence.

We will be plotting the residuals:

	Actual Values	Fitted Values	Residuals
0	91	85.089384	5.910616
1	94	83.921711	10.078289
2	0	49.608022	-49.608022
3	83	73.028155	9.971845
4	94	103.092234	-9.092234

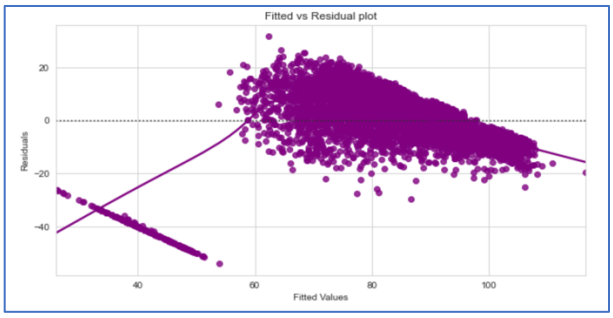


Figure 65: fitted vs residual plot

From the plot we can see a not linearity in the data, to check where the non-linearity – we will do the scatterplot of all the variable. However, the scatter has no sign of non-linearity is identified in the scatterplot. Therefore, there is no scope for transformation.

Secondly, we check for Normality:

We check the if the residuals are normally distributed.

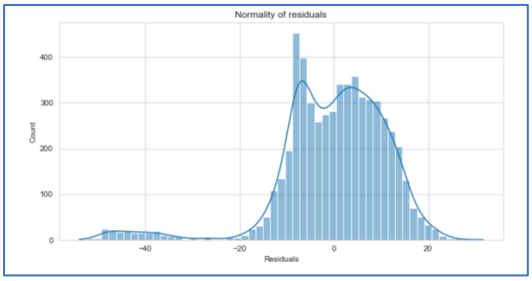


Figure 67: Normal distribution plot

From the Fig 39, we can understand that the Residuals are not entirely in a normal distribution and it is left sided tailed.

QQ Plot:

The normality assumption can be visually verified using the QQ plot of residuals. The residuals' normal probability plot should roughly resemble a straight line. However, We can see in the fig, not all the plot lying on the straight, we could notice a pattern.

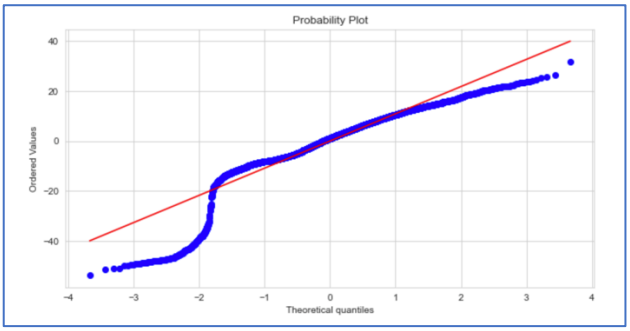


Figure 68:QQ plot

We can also check for the Shapiro-wilk test to check the normality.

Null Hypothesis: Data is normally distributed.
Alternative Hypothesis: Data is not normally distributed.

```
ShapiroResult(statistic=0.893683910369873, pvalue=0.0)
```

Since p- value is < 0.05, the residuals are not a normal distribution.

Our Final assumption would be test for Homoscedasticity:

Homoscedasticity is to check the variance in the residual are stable or constant. So we will run a hypothesis of the goldfeldquandt test are as follows:

Null hypothesis : Residuals are homoscedastic
Alternate hypothesis : Residuals have hetroscedasticity

```
[('F statistic', 1.2361488558817442), ('p-value', 7.61109260933759e-09)]
```

Since p-value < 0.05 we can say that the residuals are hetroscedasticity.

The data is referred to as heteroscedastic if the variance for the residuals across the regression line is uneven. In this situation, the residuals could take the form of an arrow or any other asymmetrical shape.

After the Assumption, we can conclude that only the no- strong multicollinearity is satisfied, where the linearity, Independence, normality and Homoscedasticity is not satisfied.

Regression Model equation:

(47.93) * Intercept + (-0.06) * lwrite + (0.0) * scall + (-0.0) * rchar + (-0.0) * wchar + (1.83) * atch + (-0.05) * pflt + (0.0) * freeswap + (6.53) * runqsz_Not_CPU_Bound

1.4 Inference: Basis on these predictions, what are the business insights and recommendations.

Business insight and Recommendation:

- R-squared of the model is 0.593, and adjusted R-squared is 0.593, which shows that the model can explain ~59% variance in the data, which is considered to be moderate.
- Null values and Duplicates has treated. In addition to that, the outlier is also treated since linear regression is sensible for outliers
- Sklearn is used for optimization, and statsmodel is used for the interpretation of the model. Dimension reduction has been made from 22 variables to 10 variables in the final model of linear regression.
- A unit increase in the 'scall' will result in a 0.0014 unit increase in the usr, all other variables remaining constant.
- A unit increase in the 'atch' will result in a 1.8333 unit increase in the usr, all other variables remaining constant.
- Data will be spilt into train and test in a 70:30 ratio. 70% of the data would be for training, and 30% would be for testing using sk.learn model.
- The Residuals are not normally distributed, the confidence interval is wide. Since the confidence interval is not stable, it is challenging to estimate the coefficient based on the minimization of least squares.
- The final model hasn't satisfied linearity, Independence, normality and Homoscedasticity assumption and is satisfied only with no strong Multicollinearity.
- The RMSE on the train and test sets are similar. Thus, overfitting is not an issue with our model.
- We can conclude the final model should be further analyzed as R square could vary from case to case.
- Moreover, from the final model, it is clear that the unstable confidence interval should be studied thoroughly to make a better model.

Problem 2: Logistic Regression, LDA and CART

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.

Dataset for Problem 2: Contraceptive_method_dataset.xlsx

Data Dictionary:

1. Wife's age (numerical)
2. Wife's education (categorical) 1=uneducated, 2, 3, 4=tertiary
3. Husband's education (categorical) 1=uneducated, 2, 3, 4=tertiary
4. Number of children ever born (numerical)
5. Wife's religion (binary) Non-Scientology, Scientology
6. Wife's now working? (binary) Yes, No
7. Husband's occupation (categorical) 1, 2, 3, 4(random)
8. Standard-of-living index (categorical) 1=verlow, 2, 3, 4=high
9. Media exposure (binary) Good, Not good
10. Contraceptive method used (class attribute) No,Yes

2.1 Data Ingestion: Read the dataset. Do the descriptive statistics and do null value condition check, check for duplicates and outliers and write an inference on it. Perform Univariate and Bivariate Analysis and Multivariate Analysis.

Below fig 41, shows the first five rows of the dataset:

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

Figure 70: First five rows of the dataset

Below fig 42 shows the last five rows of the dataset:

	Wife_age	Wife_ education	Husband_education	No_of_children_born	Wife_religion	Wife_Working	Husband_Occupation	Standard_of_living_index	Media_expos
1468	33.0	Tertiary	Tertiary	NaN	Scientology	Yes	2	Very High	Exposed
1469	33.0	Tertiary	Tertiary	NaN	Scientology	No	1	Very High	Exposed
1470	39.0	Secondary	Secondary	NaN	Scientology	Yes	1	Very High	Exposed
1471	33.0	Secondary	Secondary	NaN	Scientology	Yes	2	Low	Exposed
1472	17.0	Secondary	Secondary	1.0	Scientology	No	2	Very High	Exposed

Figure 72: Last five rows of the dataset

Duplicates:

```
Number of duplicate rows = 80
```

There are 80 duplicates present in the dataset, and those 80 duplicate rows have been dropped from the dataset.

Data information:

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1393 entries, 0 to 1472
Data columns (total 10 columns):
 #   Column                     Non-Null Count  Dtype
---  ------                     --------------  -----
 0   Wife_age                   1326 non-null   float64
 1   Wife_ education            1393 non-null   object
 2   Husband_education          1393 non-null   object
 3   No_of_children_born        1372 non-null   float64
 4   Wife_religion              1393 non-null   object
 5   Wife_Working               1393 non-null   object
 6   Husband_Occupation         1393 non-null   int64
 7   Standard_of_living_index   1393 non-null   object
 8   Media_exposure             1393 non-null   object
 9   Contraceptive_method_used  1393 non-null   object
dtypes: float64(2), int64(1), object(7)
memory usage: 119.7+ KB
```

Figure 74: Data Information

As we could see in the above fig 43. there are 1472 rows and 10 features. There are 10 variables in the dataset. Out of which, 7 are object datatype, 2 float datatype and one integer datatype. Furthermore, it is evident that wife_age and no_of_children_born have few missing values in the dataset.

Descriptive statistics:

	count	unique	top	freq	mean	std	min	25%	50%	75%	max
Wife_age	1326.0	NaN	NaN	NaN	32.557315	8.289259	16.0	26.0	32.0	39.0	49.0
Wife_ education	1393	4	Tertiary	515	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Husband_education	1393	4	Tertiary	827	NaN	NaN	NaN	NaN	NaN	NaN	NaN
No_of_children_born	1372.0	NaN	NaN	NaN	3.290816	2.399697	0.0	1.0	3.0	5.0	16.0
Wife_religion	1393	2	Scientology	1186	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Wife_Working	1393	2	No	1043	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Husband_Occupation	1393.0	NaN	NaN	NaN	2.174444	0.85459	1.0	1.0	2.0	3.0	4.0
Standard_of_living_index	1393	4	Very High	618	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Media_exposure	1393	2	Exposed	1284	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Contraceptive_method_used	1393	2	Yes	779	NaN	NaN	NaN	NaN	NaN	NaN	NaN

Figure 76: Descriptive statistics

The above Descriptive statistics depicts the dataset's mean, median, min, max and lower and upper quartile values. It is interesting to see many variables, min and 25%, is zero in the datasets.

Null Value check:

```
Wife_age                  67
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
dtype: int64
```

Figure 78: Null Value Check

```
Wife_age                  0
Wife_ education            0
Husband_education          0
No_of_children_born        0
Wife_religion              0
Wife_Working               0
Husband_Occupation         0
Standard_of_living_index   0
Media_exposure             0
Contraceptive_method_used  0
dtype: int64
```

Figure 77: After Imputing Null Value

From the fig 46, it is clear there are 67 null values present in the wife_age variable and 21 null values in the no_of_children_born variable. And the null value has been imputed median. The fig 45 . shows after imputing the null value.

Outlier check:

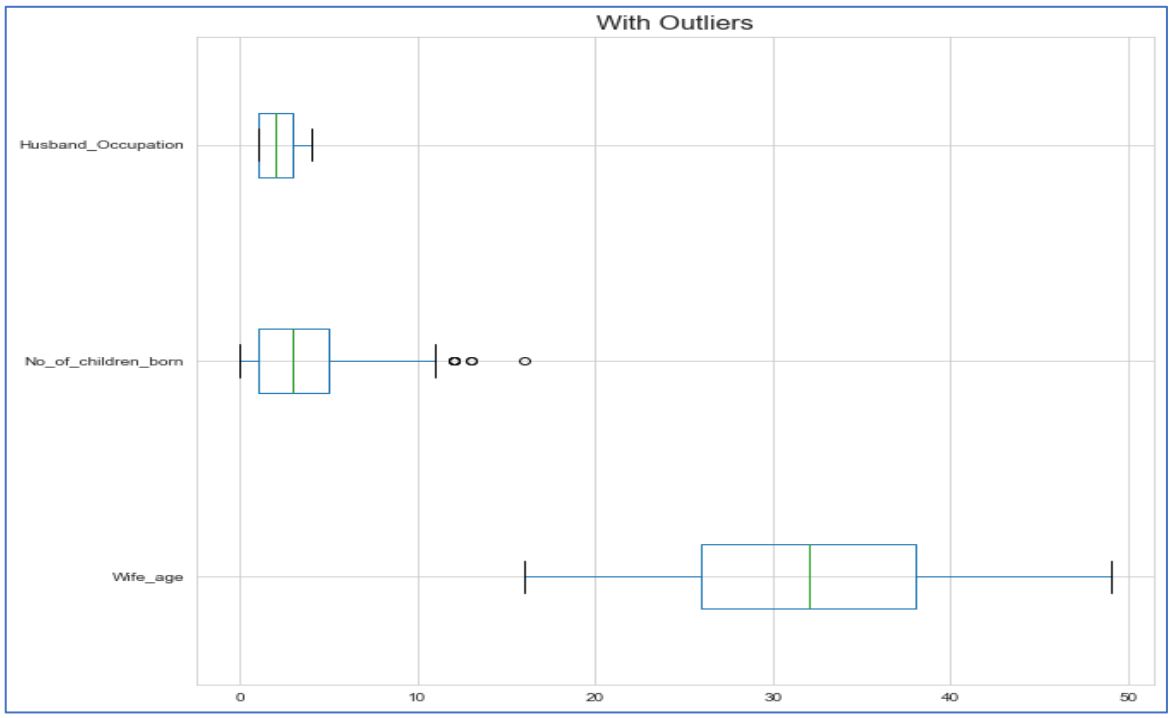


Figure 79: Outlier Check

From the above fig 47 we can understand there outlier present in the no of children born variable.

Univariate:

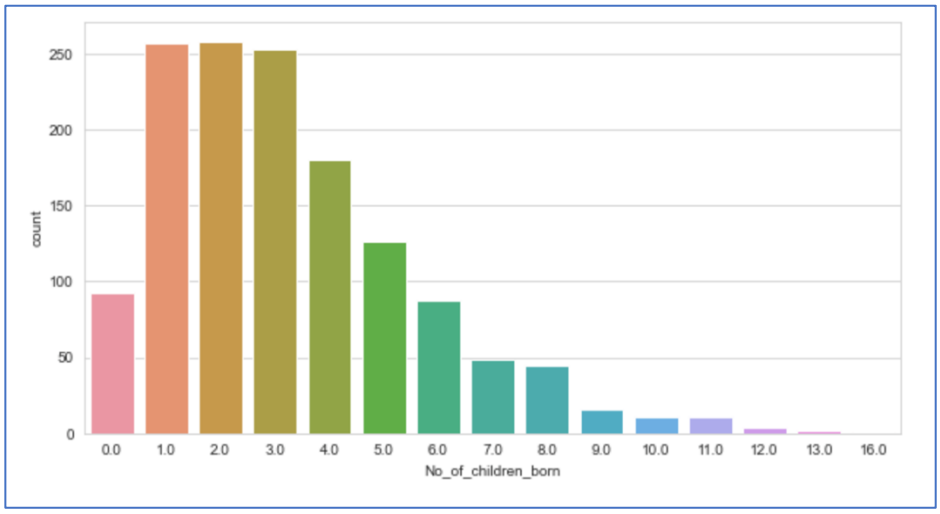


Figure 80: Univariate - No of Children born

From the above fig 48, we can understand the majority of married women have 1 to 3 children, and only a few married women have a min of 9 to 13 children.

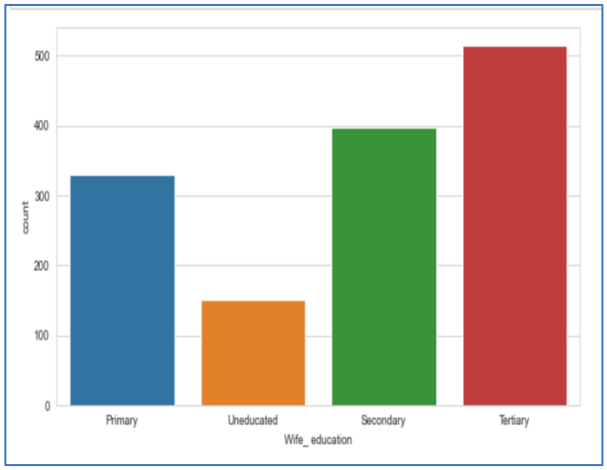


Figure 82: Univariate - Wife Education

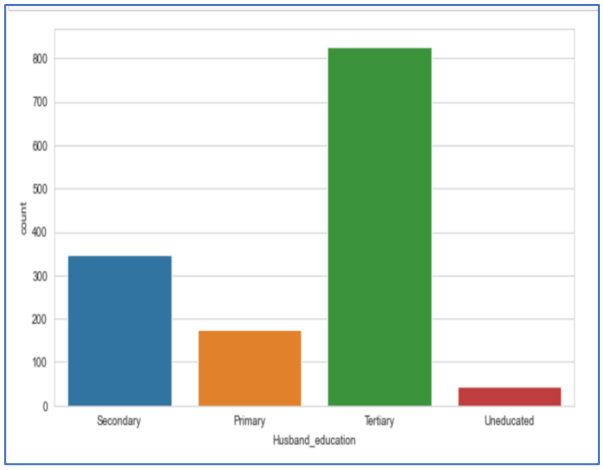


Figure 81: Univariate - Husband Education

From the fig 50 above, it is evident that the majority of women are educated, and only a few are uneducated. Similarly, the majority of the husband have completed a tertiary level of education, and only a few are uneducated. A similar pattern is maintained in both the wives' and Husband's education.

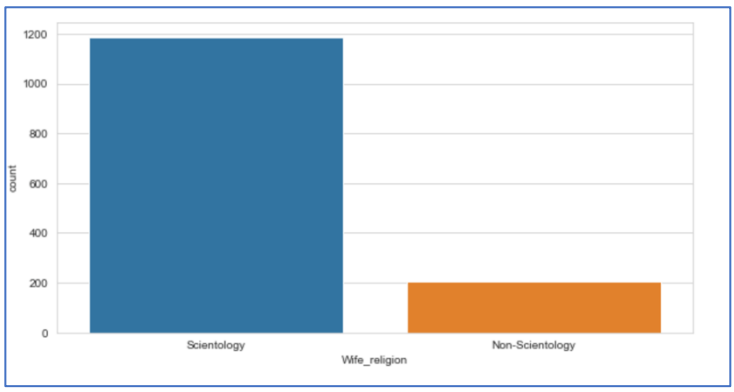


Figure 83: Univariate - Wife Religion

As we can see from the above fig 51, the preponderance of wives follow the Scientology religion compared to the Non-Scientology.

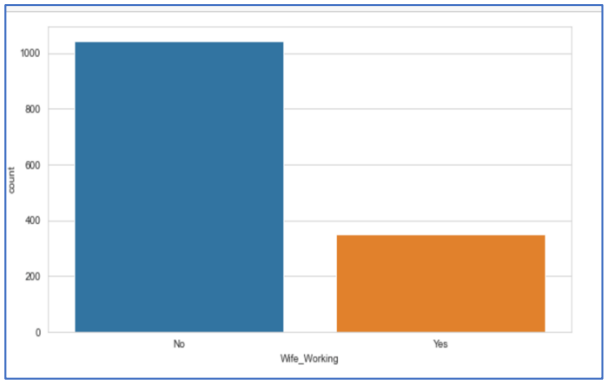


Figure 85: Univariate - Wife working

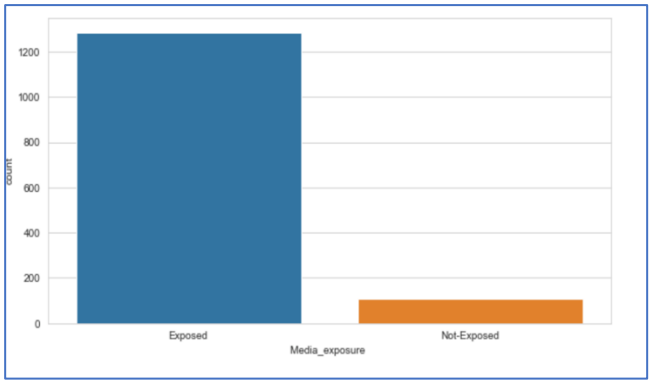


Figure 84: Univariate - Media Exposure

From the left fig 53, it is evident that most of the wives are not working.
From the right fig 52, it is clear that the majority of wives have exposure to the media, and only a fewer have no exposure.

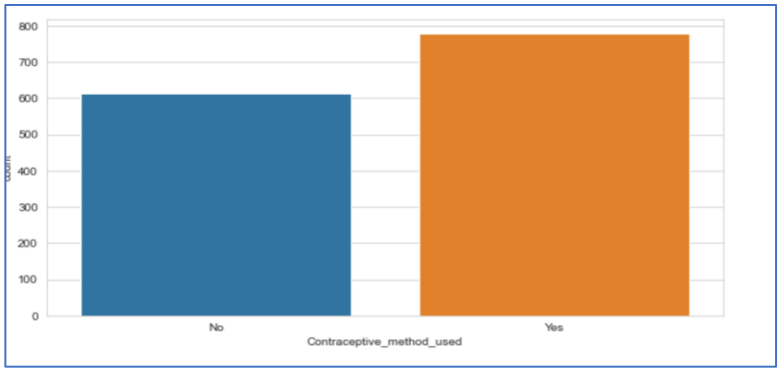


Figure 86: Univariate - Contraceptive method used

The above represents the contraceptive methods used; It is obvious most wives are using contraceptive methods compared to wives who are not using contraceptives method.

Bivariate Analysis:

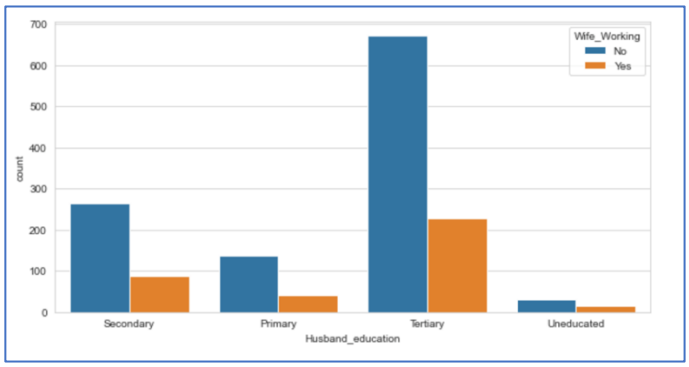


Figure 87: Bivariate - Husband Education Vs Wife Working

As we can see from the above fig 55, it is clear that most of the wives who are not working have a husband with a Tertiary level of education. On the other hand, when it comes to the working wife category – most of the wives working also have a husband with a tertiary level of education.

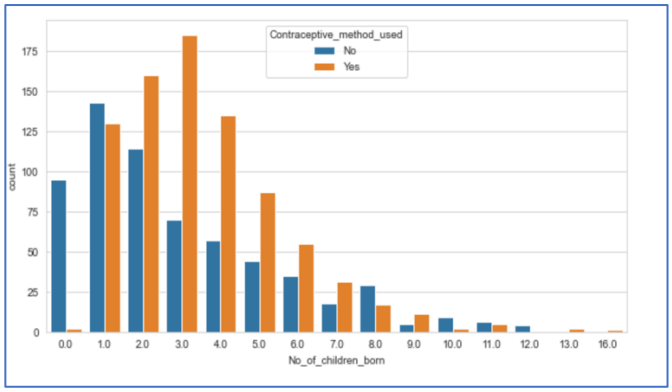


Figure 88: Bivariate - No of Children born Vs Contraceptive method used

As we could see in the above fig 56, it is evident that most of the family who have 3 to 7 children's use the contraceptive method. It is surprise to see, the family who have no kids don't use contraceptive method.

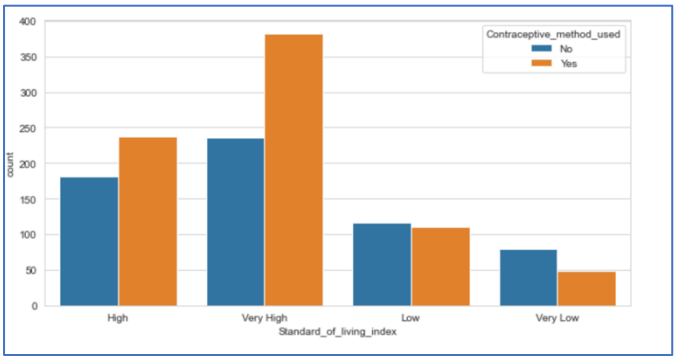


Figure 89: Bivariate - Standard of living index Vs Contraceptive method used

The above figure 57, depicts the standard of the living index vs the contraceptive method. And the wives who use the contraceptive method have a high standard live index, whereas those who fall under a low or very low standard of living index don't use the contraceptive method.

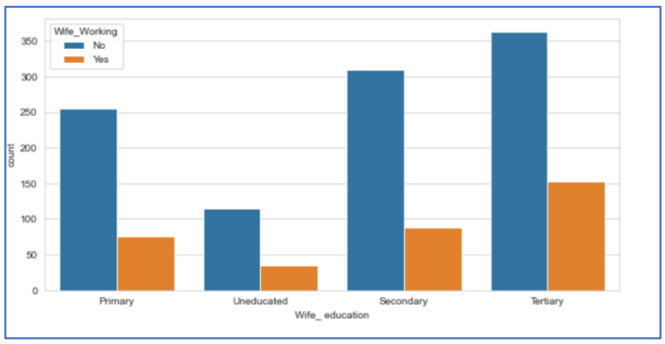


Figure 90: Bivariate - Wife Education Vs Wife Working

The above figure 58, shows the wife's education vs the wife working; as we can see, most wives who have completed their tertiary education are not working compared to those who are working. A similar pattern is maintained through this category.

Multivariate Analysis:

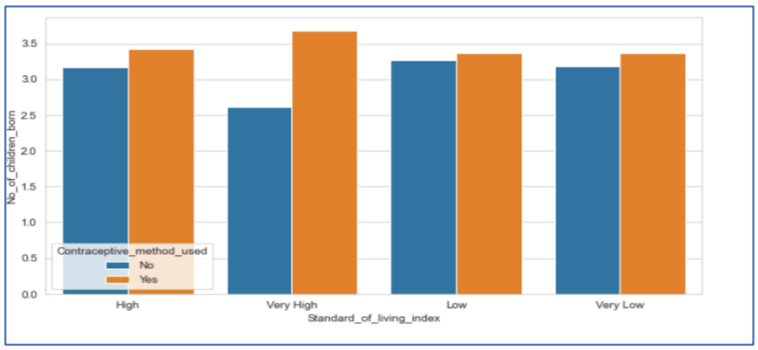


Figure 91: Multivariate - Standard of living Index vs No for children born vs Contraceptive method used

As is clear from the fig 59, the wives whose standard of living index is high with more children are using contraceptive methods. On the other hand, for wives whose standard of living is low or very low and who have an average of more than three kids, the usage of contraceptive methods has only a marginal difference. However, the use of contraceptive methods is slightly high in this case.

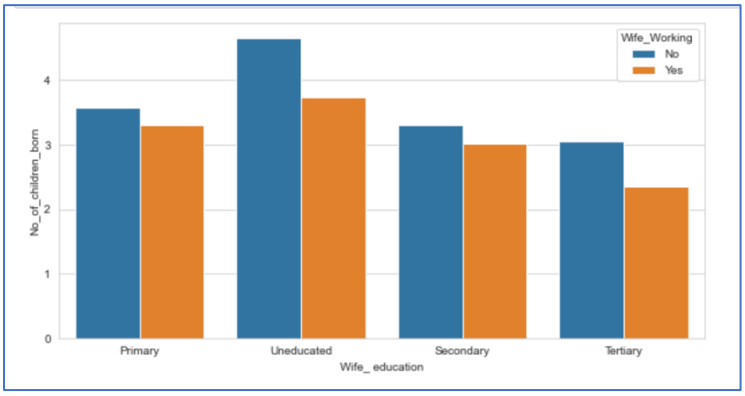


Figure 92: Multivariate - Wife Education Vs No of children born Vs wife working

As evident in the above fig 60, we can see that uneducated wives with more than four children are not working. We can see a pattern maintained throughout the education level; wives who are not working have more children than wives who are working.

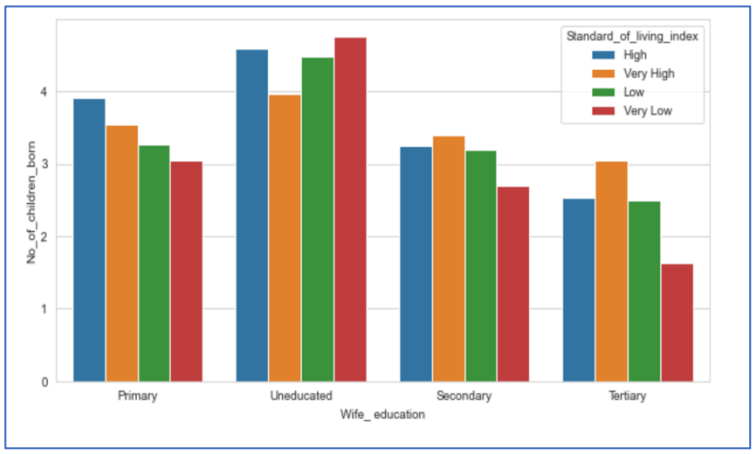


Figure 93: Multivariate - Wife education vs No of children born vs standard of living index

The above bar chart represents the wife's education, No of children born and their standard of living index. The uneducated wives who have more number children have a low standard of living index. The wives with tertiary level of education with average kids of 3 have a high standard of living index.

2.2 Do not scale the data. Encode the data (having string values) for Modelling. Data Split: Split the data into train and test (70:30). Apply Logistic Regression and LDA (linear discriminant analysis) and CART.

1. Wife's education (categorical) 1=uneducated, 2= primary, 3= Secondary, 4=tertiary
2. Husband's education (categorical) 1=uneducated, 2= primary, 3= Secondary, 4=tertiary
3. Standard-of-living index (categorical) 1=very low, 2=low, 3=High, 4= Very high

We will be encoding the wife's education, Husbands education and standard of living index in an ordinal manner as mentioned in the above order.

Furthermore, the above mentioned 3 variable will be converted from object to integer datatype.

Encoding the string based values using pd.get_dummies:

Dummies encoding will be done for the string-based variable so that the two columns are not linearly dependent on one another. Furthermore, a new variable will be created to store the encoded data.

Fig show first five rows of the dataset after the dummy columns were created:

_of_children_born	Husband_Occupation	Standard_of_living_index	Wife_religion_Scientology	Wife_Working_Yes	Media_exposure _Not-Exposed	Contraceptive_method_used_Yes
3.0	2	3	1	0	0	0
10.0	3	4	1	0	0	0
7.0	3	4	1	0	0	0
9.0	3	3	1	0	0	0
8.0	3	2	1	0	0	0

Figure 94: First five rows of the dummy variable columns

Before, Splitting the data we will be separating the x and y since the independent feature will be only used to train the model; it will be used to predict the target variable.

X variable will be created by including only the independent variable, Therefore, target variable will be dropped –' ***Contraceptive_method_used_Yes'***.

Y variable will be created by only keeping the target variable which is '***Contraceptive_method_used_Yes'***

Wife_ ducation	Husband_education	No_of_children_born	Husband_Occupation	Standard_of_living_index	Wife_religion_Scientology	Wife_Working_Yes	Media_exposure _Not-Exposed
2	3	3.0	2	3	1	0	0
1	3	10.0	3	4	1	0	0
2	3	7.0	3	4	1	0	0
3	2	9.0	3	3	1	0	0
3	3	8.0	3	2	1	0	0

Figure 95: First five rows of X variable

```
0    0
1    0
2    0
3    0
4    0
Name: Contraceptive_method_used_Yes, dtype: uint8
```

Figure 96: First five rows of Y Variable

Train – Test Spilt

Data will spilt into train and test in 70:30 ratio. 70% of the data would be for training and 30% will be for testing. Stratify parameter will be used will splitting the data to ensure that train and test has same proportion of 0 and 1 as actual data.

a) Applying logistic regression model:

Firstly, we will initiate the **logistic regression model** and will fit the training data to it.

```
LogisticRegression(max_iter=10000, penalty='none', solver='newton-cg',
                   verbose=True)
```

Further, predicting will be done using predict method.

Prediction for testing data:

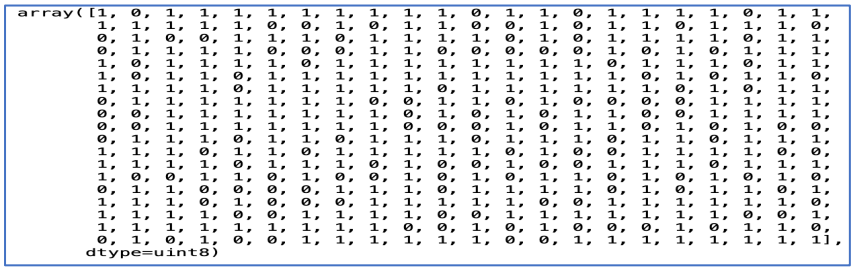


Figure 97: Prediction for testing data

Predicting the probability of the test data:

	0	1
0	0.282220	0.717780
1	0.614046	0.385954
2	0.335348	0.664652
3	0.297914	0.702086
4	0.244837	0.755163

Figure 98: Predicting the probability of the test data

b) Initiating the LDA Model

Secondly, we build the LDA Model using the Linear Discriminant Analysis() and training data will be fit into it.

Generating the coefficient and intercept for the Linear Discriminant Function:

The intercept value for the LDA is `array([-0.81058333])`

And the coefficient of Linear Discriminant function is

```
[[-0.0731937 ,  0.51534689,  0.03615321,  0.31062619,  0.13736477,
   0.31957476, -0.44996625, -0.17085182, -0.34267426]])
```

Further, predicting will be done using predict method for the test and train data.

Prediction for testing data:

```
[1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1,
 1, 1, 1, 1, 1, 0, 0, 1, 0, 1, 1, 0, 1, 1, 0, 1, 1, 0, 1, 1, 1, 0,
 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 0, 1, 1,
 0, 1, 1, 1, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 1, 1,
 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1,
 1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 0,
 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1,
 0, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 0, 1, 1, 0, 0, 1, 1, 1, 1, 1,
 0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 0, 1, 1, 0, 0, 1, 1, 1, 1,
 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 0, 1, 1, 0, 1, 0, 1, 0, 0,
 0, 1, 1, 1, 0, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 0, 1, 1, 1,
 1, 1, 1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 0, 0, 1, 1, 1, 0, 1, 0, 0,
 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 0, 0, 1, 0, 0, 1, 1, 1, 0, 1, 1, 1,
 1, 0, 0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0, 1, 0, 1, 1,
 0, 1, 1, 0, 0, 0, 0, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 0, 1, 1, 0, 1,
 1, 1, 1, 0, 1, 0, 0, 0, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 0,
 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1,
 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 0, 0, 0, 1, 0, 1, 1, 0,
 0, 1, 0, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1],
dtype=uint8)
```

Figure 99: Prediction for testing data

c) Initiating the CART:

CART model is build using the decision tree classifier, and the criterion within the decision tree classifier is “gini”, and then the training data will be fit into it. Once the model is trained, before checking the prediction- we can visualise the decision tree to see how the decision tree has spilt at various interval and checking parent node, root node, internal node observation and the split criteria.

Sklearn is used to view the graphical visualisation function which is available sklearn Tree submodule.

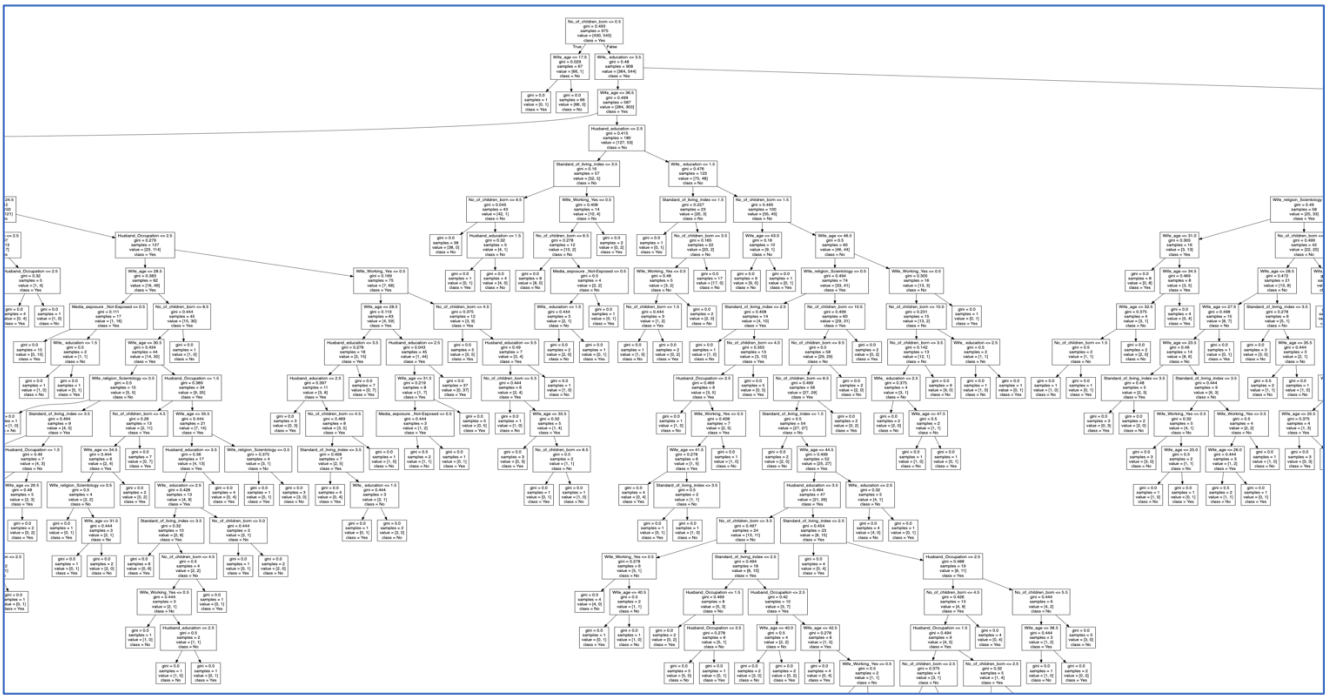


Figure 100: Graphical visualisation of the Decision Tree

As we see in the fig, the decision tree is overgrown, and the terminal node has only a few sample records. So most of the terminal nodes don't have an adequate number of samples. That indicates the decision tree has grown to its full depth. Therefore it needs to be pruned.

Now, we will pass in some tuning parameter as below into the Decision Tree Classifier.

```
DecisionTreeClassifier(max_depth=8, min_samples_leaf=100, min_samples_split=100,
                       random_state=1)
```

Regularising the Decision Tree:

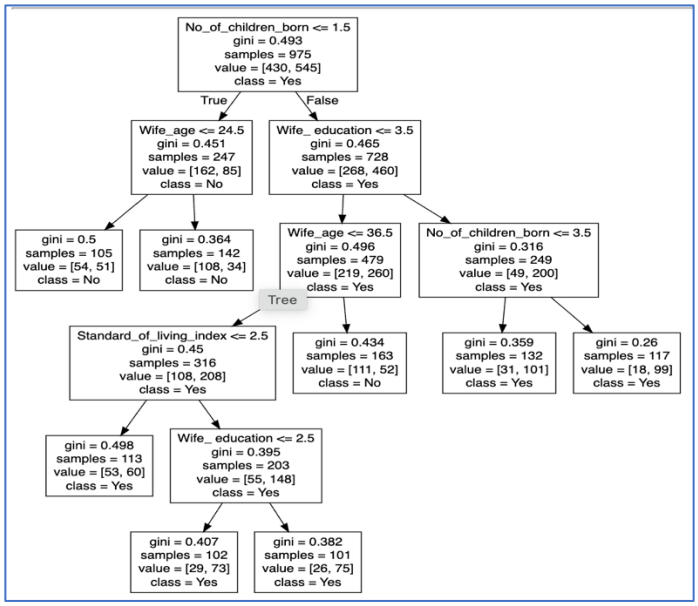


Figure 101: Regularising the Decision Tree:

From the fig 69. we can notice depth level has reduced significantly after the decision tree is regularised. Now we have built the model, we will extract the feature importance (Fig 70)

	Imp
Wife_age	0.350811
No_of_children_born	0.342979
Wife_ education	0.243889
Standard_of_living_index	0.062322
Husband_education	0.000000
Husband_Occupation	0.000000
Wife_religion_Scientology	0.000000
Wife_Working_Yes	0.000000
Media_exposure _Not-Exposed	0.000000

Figure 102: Feature importance

After building the decision tree model, the next step is prediction. Predicting will be done using predict method for the test and train data.

Prediction for test data:

```
array([0, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 0, 1, 1,
       1, 1, 1, 1, 0, 0, 1, 0, 1, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 0, 1, 1,
       1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 0, 1, 0, 1, 0, 1, 1, 1, 1, 1, 0, 0, 1,
       0, 1, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 1, 1, 1, 1, 0, 0, 0, 1, 0, 1,
       1, 0, 0, 1, 1, 0, 0, 1, 0, 1, 0, 0, 1, 1, 1, 1, 0, 1, 1, 0, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 0, 1, 1, 0,
       0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1,
       1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 0, 1, 0, 0, 0, 1, 1, 0, 0, 0,
       0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 0, 1, 1, 0, 1, 0, 0, 1, 1,
       0, 1, 1, 0, 1, 1, 1, 0, 1, 0, 0, 0, 1, 0, 1, 1, 0, 0, 0, 1, 1, 1,
       0, 0, 1, 1, 0, 1, 0, 0, 0, 1, 1, 1, 0, 1, 0, 0, 1, 1, 0, 1, 1, 0,
       1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 0, 0, 1, 1, 0, 1, 0, 0,
       1, 0, 1, 1, 0, 1, 1, 1, 0, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0, 1, 1, 1,
       0, 0, 1, 0, 1, 0, 1, 0, 1, 1, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 1, 0,
       0, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 1, 1,
       1, 0, 1, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 1, 0, 1, 1, 1, 1, 0,
       1, 0, 1, 1, 0, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 0, 1, 1, 0, 0,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 0, 0, 1, 0, 1, 1, 1, 0, 0,
       0, 1, 1, 1, 0, 0, 1, 0, 0, 0, 1, 1, 0, 1, 1, 0, 0, 1, 0, 1, 1, 1],
      dtype=uint8)
```

Figure 103:Prediction for test data

Predicting the probability of the test data:

	0	1
0	0.514286	0.485714
1	0.514286	0.485714
2	0.257426	0.742574
3	0.234848	0.765152
4	0.234848	0.765152

Figure 104: Predicting the probability of the test data

2.3 Performance Metrics: Check the performance of Predictions on Train and Test sets using Accuracy, Confusion Matrix, Plot ROC curve and get ROC_AUC score for each model Final Model: Compare Both the models and write inference which model is best/optimized.

From the sklean method we will import the classification report, confusion matrix and ROU_AUC score. By using these 2 function we can create reports for test and train samples.

Confusion Matrix: The confusion matrix helps to understand where the error is made in the model. The row shows the actual outcome, and the columns represent the prediction. This method helps to know where the predictions have gone wrong.

Classification Report: Classification report helps to measure the quality of the predication. There are three important matrix in the classification report.

1. Accuracy
2. Recall
3. Precision
4. F1 score

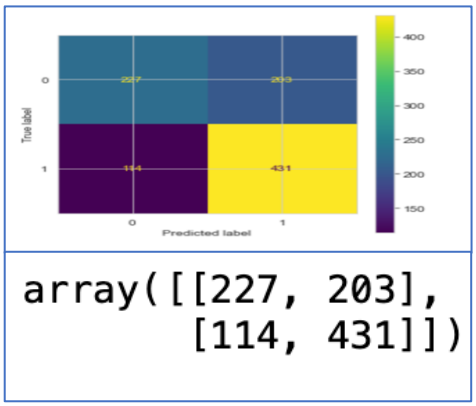
In the reports, the higher the accuracy stronger the model. Recall denotes how many Actual true data points are identified as true data. And Precision explains the measure of the positive prediction made by the model. The F1 score shows the combination of recall and Precision in the model.

ROC_AUC score: ROC curve that is closer to the top left denotes the best performance of the model. And the AUC score range from 0 to 1, O indicates wrong prediction and 1 represents 100% correct predication.

Logistic Regression Model:

We will create confusion matrix , classification report and ROU_AUC score for the training and testing data is created. Therefore, we feed in the actual and predicted data for both the function.

Confusion matrix for Training Data & Test Data:



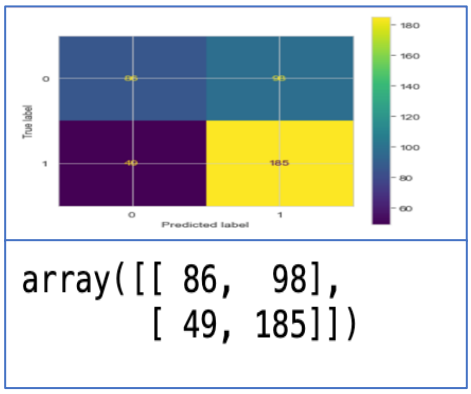


Figure 105: ***Logistic Regression Model*** *- Confusion matrix for Training Data & Test Data:*

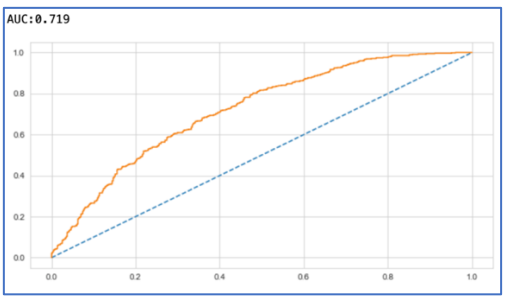
Classification Report for Training data and Test data:

	precision	recall	f1-score	support
0	0.67	0.53	0.59	430
1	0.68	0.79	0.73	545
accuracy			0.67	975
macro avg	0.67	0.66	0.66	975
weighted avg	0.67	0.67	0.67	975

	precision	recall	f1-score	support
0	0.64	0.47	0.54	184
1	0.65	0.79	0.72	234
accuracy			0.65	418
macro avg	0.65	0.63	0.63	418
weighted avg	0.65	0.65	0.64	418

Figure 106: ***Logistic Regression Model*** *Classification Report for Training data and Test data*

ROC_AUC Curve: For Training data & Test Data:



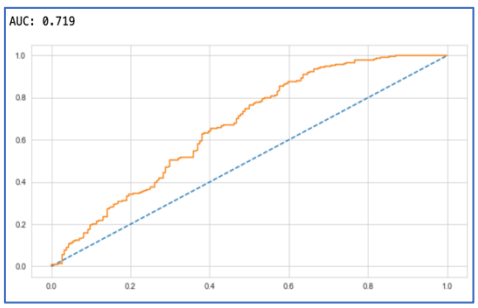


Figure 107: ***Logistic Regression Model-*** *ROC_AUC Curve: For Training data & Test Data*

AUC for Training Data: 0.719
AUC for Testing Data: 0.719

LDA Model:

We will create confusion matrix , classification report and ROU_AUC score for the training and testing data is created. Therefore, we feed in the actual and predicted data for both the function.

Confusion matrix for Training & Testing Data:

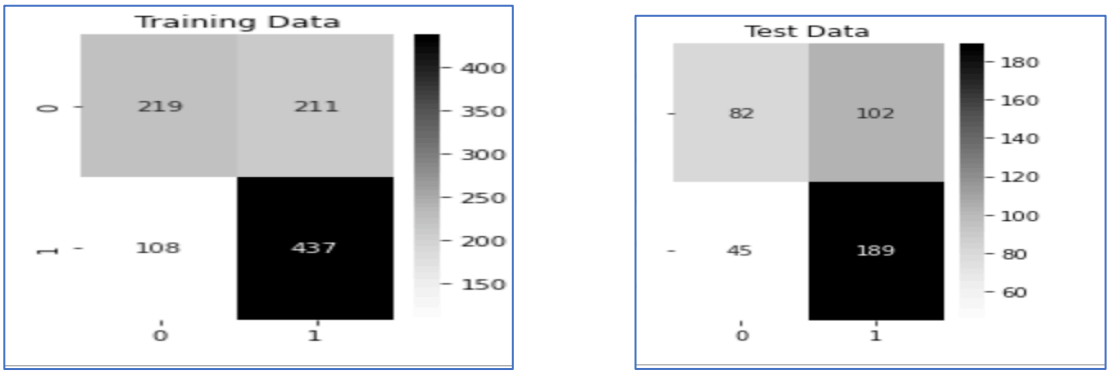


Figure 108: LDA MODEL _ Confusion matrix for Training & Testing Data

Classification Report for Training and Test Data:

Classification Report of the training data:

	precision	recall	f1-score	support
0	0.67	0.51	0.58	430
1	0.67	0.80	0.73	545
accuracy			0.67	975
macro avg	0.67	0.66	0.66	975
weighted avg	0.67	0.67	0.66	975

Classification Report of the test data:

	precision	recall	f1-score	support
0	0.65	0.45	0.53	184
1	0.65	0.81	0.72	234
accuracy			0.65	418
macro avg	0.65	0.63	0.62	418
weighted avg	0.65	0.65	0.64	418

Figure 109: LDA MODEL - Classification Report for Training and Test Data

ROC_AUC Curve: For Training & Test Data:

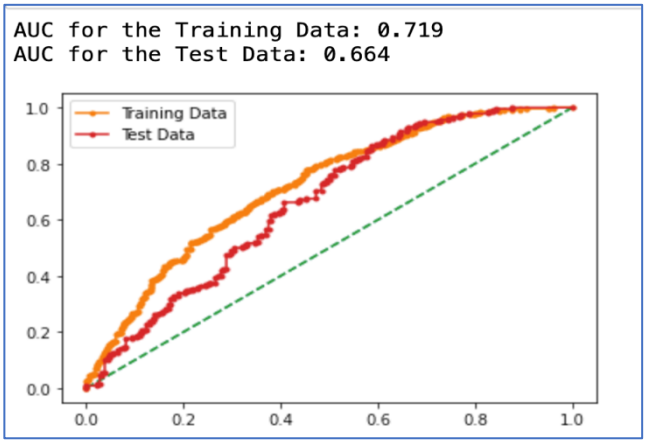


Figure 110: LDA MODEL - ROC_AUC Curve: For Training & Test Data

AUC for the Training Data: 0.719
AUC for Test Data: 0.664

CART Model: (Classification and Regression Trees Model)

We will create confusion matrix , classification report and ROU_AUC score for the training and testing data is created. Therefore, we feed in the actual and predicted data for both the function.

Confusion matrix for Training Data & Testing Data:

```
array([[273, 157],
       [137, 408]])
```

```
array([[115,  69],
       [ 57, 177]])
```

Figure 111: CART Model - Confusion matrix for Training Data & Testing Data

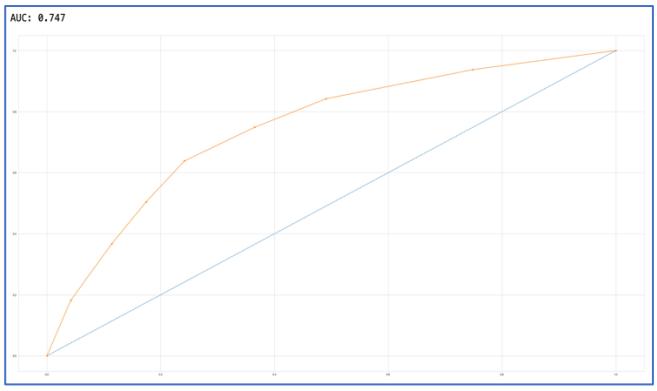
Classification Report for Training and Test Data:

	precision	recall	f1-score	support
0	0.67	0.63	0.65	430
1	0.72	0.75	0.74	545
accuracy			0.70	975
macro avg	0.69	0.69	0.69	975
weighted avg	0.70	0.70	0.70	975

	precision	recall	f1-score	support
0	0.67	0.62	0.65	184
1	0.72	0.76	0.74	234
accuracy			0.70	418
macro avg	0.69	0.69	0.69	418
weighted avg	0.70	0.70	0.70	418

Figure 112: CART Model - Classification Report for Training and Test Data

ROC_AUC Curve: For Training & Test Data:



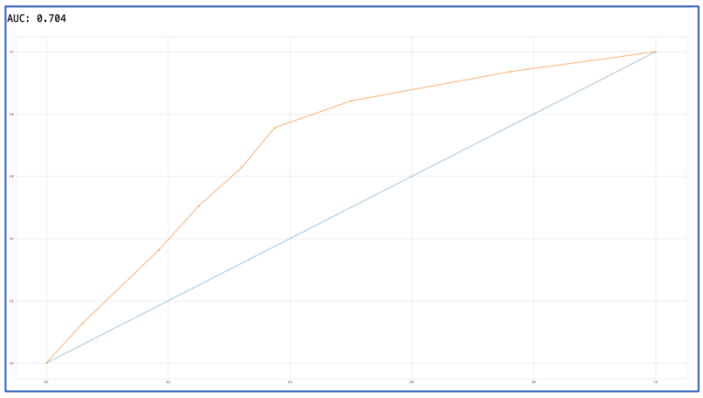


Figure 113: CART Model - ROC_AUC Curve: For Training & Test Data

AUC for the Training Data: 0.747
AUC for Test Data: 0.704

Inference: Logistic Regression Model

The wives who are using the Contraceptive method (Label 1) is our class of interest. Additionally, while comparing the classification report and ROC_AUC Curve of both the training and testing data, the prediction seems to be the same. The performance of the logistic regression model is stable. To solve this problem, let's assume a benchmark for this problem of recall is 80%. Recall should be considered when the goal of the model is to detect all relevant instances in a dataset, regardless of the number of false positives produced. From the classification report, the recall is at 0.79, which is closer to 0.80.

AUC for Training Data: 0.719
AUC for Testing Data: 0.719

So, in the context of false negatives, recall would be a more appropriate metric to consider as it directly measures the ability of the model to avoid false negatives. Recall (79%) for test data – Out of all the Customers who actually did used Contraceptive method, 79% of Customers did used Contraceptive method have been predicted correctly.

The model **Accuracy** is on training data is 67% and testing data is 65%. However, we won't be able to consider the accuracy since there is class imbalance.

Inference: LDA Model

As we could see in the Linear Discriminant Analysis model, while comparing the classification and AUC score among the train and test data, the prediction of the classification report is close. However, the prediction of AUC sore is note the same.

AUC for the Training Data: 0.719
AUC for Test Data: 0.664

The wives who are using the Contraceptive method (label -1) is our class of interest.

So, in the context of false negatives, recall would be a more appropriate metric to consider as it directly measures the ability of the model to avoid false negatives. Recall (81%) for test data– Out of all the Customers who actually did used Contraceptive method, 81% of Customers did used Contraceptive method have been predicted correctly.

Accuracy of Test data: 67; **Accuracy of testing data**:65. Accuracy can't be considered due to the class imbalance.

When the Training and test datasets is the same but the AUC score is not, it suggests that the model is overfitting on the training dataset.

Inference: CART Model- Classification and Regression Tree

As we could see in the CART model, while comparing the classification and AUC score among the train and test data, the prediction of the classification report is same. However, the prediction of AUC sore is note the same.

AUC for the Training Data: 0.747
AUC for Test Data: 0.704

The wives who are using the Contraceptive method (Label 1) is our class of interest.

So, in the context of false negatives, recall would be a more appropriate metric to consider as it directly measures the ability of the model to avoid false negatives. **Recall for test data** – Out of all the Customers who actually did used Contraceptive method, 76% of Customers did used Contraceptive method have been predicted correctly.

Accuracy of Test data: 70; **Accuracy of testing data**:70. Accuracy can't be considered due to the class imbalance.

When the Training and test datasets is the same but the AUC score is not, it suggests that the model is overfitting on the training dataset.

Best model:

After analysing all three models (Logistic Regression Model, LDA and CART model), it is evident that the Logistic regression model is a perfect fit since the Logistic regression model has a similar classification report (recall metrics), and the AUC score for both the training and test is same. Therefore, that clearly indicates the model is not overfitting, and the logistic regression model has the best performance in both train and test data, so it is clear that model should generalize well in the unseen or new data.

2.4 Inference: Basis on these predictions, what are the insights and recommendations.

Business Insight:

- The dataset had few duplicates and Null value where duplicates has been dropped and the null values has been imputed using the median method.
- There are 1472 rows and 10 features in the dataset.
- All the string values has been encode to integer datatype.
- From the Exploratory data analysis, it is clear that most of the wives are using the Contraceptive method.
- We can see a pattern maintained throughout the education level; wives who are not working have more children than wives who are working.
- The wives who use the contraceptive method have a high standard live index ad those are from very low living index comparatively don't use contraceptive method. Therefore it is evident that ***socio-economic characteristics*** influence the usage of the contraceptive method.

- The recall of Logistic regression model is only 79%, but it can always be improved. However, After examining each model, it is evident that the logistic regression model performs the best in both train and test data, indicating that it should adapt well to a new or unknown set of data.