PROJECT
ON
PREDICTIVE MODELLING

By SHAJIL FERNANDEZ

09-09-2023

Table of Contents:

Problem 1

Problem 2

List of Figures:

List of Tables:

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.

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**.**

Ans:

- **Table 1: Top 5 rows of the dataset**

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

- **Table 2: Bottom 5 rows of the dataset**

lread	lwrite	scall	sread	swrite	fork	exec	rchar	wchar	pgout	...	pgscan	atch	pgin	ppgin	pflt	vflt	runqsz	freemem	freeswap	usr
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	986647	80
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	1055742	90
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	969106	87
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	1022458	83
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	1756514	94

- There are 8192 rows and 22 columns in the given data.

```
There are 8192 rows and 22 columns.
```

- **Info:**

Table 3: Basic info of the dataset

```
<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)
```

Observations:

- There are 8192 rows and 22 columns.

- There are 13 float64, 8 int64 and 1 object datatypes.

- There are missing values for 2 columns (rchar and wchar).

- There are **no duplicate rows** in the given dataset.

- **Null values:**

Table 4: Table showing missing value info in the dataset

```
lread          0
lwrite         0
scall          0
sread          0
swrite         0
fork           0
exec           0
rchar        104
wchar         15
pgout          0
ppgout         0
pgfree         0
pgscan         0
atch           0
pgin           0
ppgin          0
pflt           0
vflt           0
runqsz         0
freemem        0
freeswap       0
usr            0
dtype: int64
```

Observations:

- There are 104 missing values in rchar column and 15 missing values in wchar column.

- **5-point summary of numerical and categorical data:**

Table 5: Table showing 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

	count	unique	top	freq
runqsz	8192	2	Not_CPU_Bound	4331

Observations:

- Reads (transfers per second) between system memory and user memory ranges from minimum of 0 to maximum of 1845.

- Writes (transfers per second) between system memory and user memory ranges from minimum of 0 to maximum of 575.

- About 4331 of 8192 are ‘Not_CPU_Bound’.

- Portion of time (%) that cpus run in user mode ranges from minimum of 0 to maximum of 99 and about 75% of the usr fall under 94.

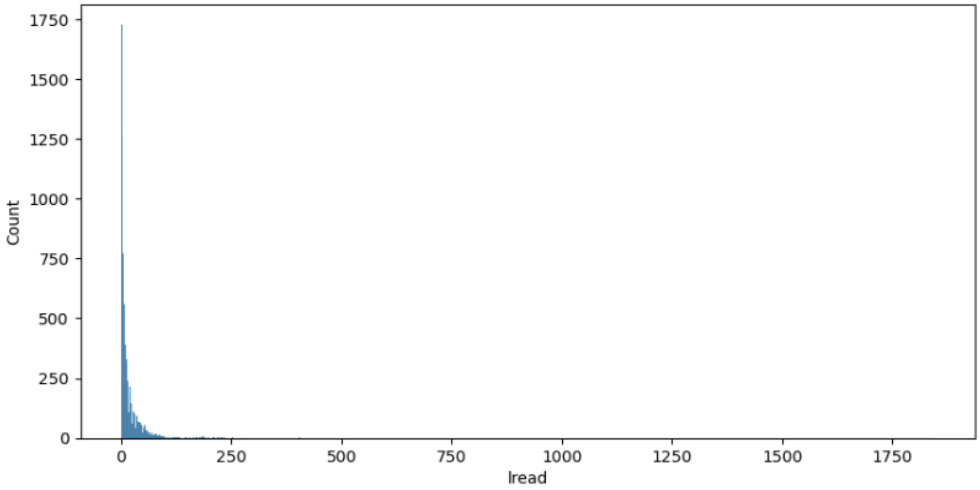
- Unique values of categorical data:

```
runqsz
 Not_CPU_Bound    4331
CPU_Bound         3861
Name: runqsz, dtype: int64
```

- **Univariate Analysis:**

a) **Lread**:

Figure 1: Histplot showing count of lread



Statistical summary of the variable

```
count    8192.000000
mean       19.559692
std        53.353799
min         0.000000
25%         2.000000
50%         7.000000
75%        20.000000
max      1845.000000
Name: lread, dtype: float64
```

Observations

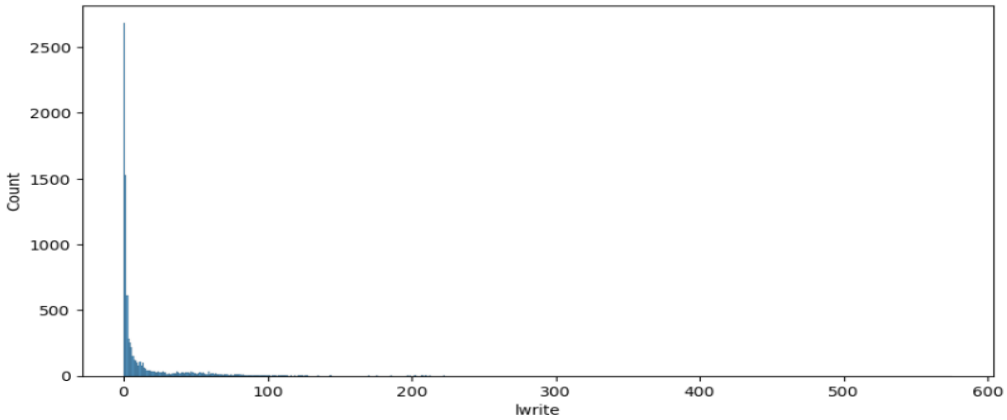
- The average number reads between system memory and user memory is 19.56.

- lread ranges from 0 to 1845.

- About 75% of the lread fall under 20.

b) **Lwrite**:

Figure 2: Histplot showing count of lwrite



Statistical summary of the variable

```
count    8192.000000
mean       13.106201
std        29.891726
min         0.000000
25%         0.000000
50%         1.000000
75%        10.000000
max       575.000000
Name: lwrite, dtype: float64
```

Observations

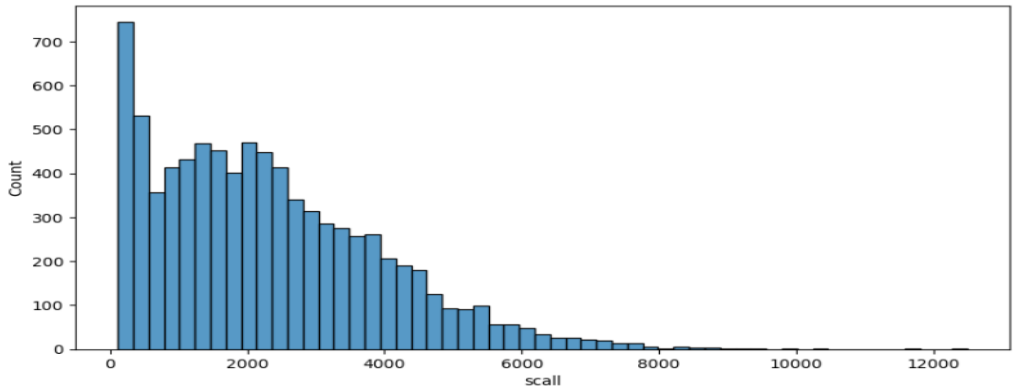
- The average number writes between system memory and user memory is 13.11.

- lwrite ranges from 0 to 575.

- About 75% of the lwrite fall under 10.

c) **scall**:

Figure 3: Histplot showing count of scall



Statistical summary of the variable

```
count     8192.000000
mean      2306.318237
std       1633.617322
min        109.000000
25%       1012.000000
50%       2051.500000
75%       3317.250000
max      12493.000000
Name: scall, dtype: float64
```

Observations

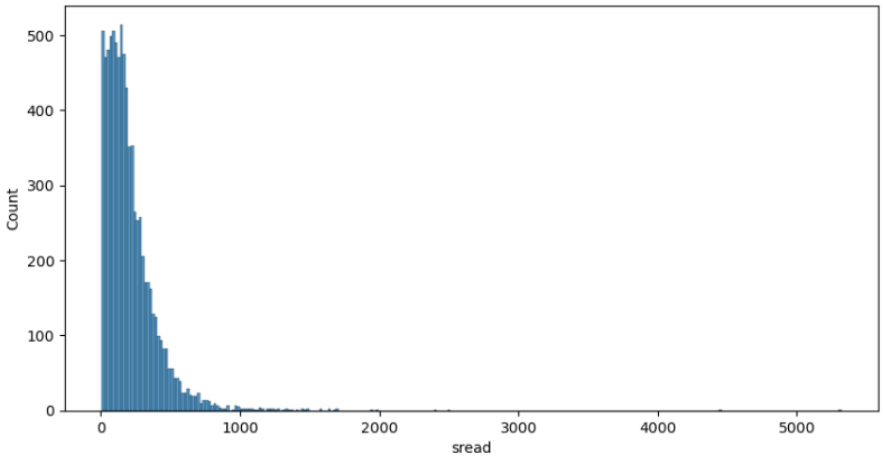
- The average number of system calls of all types per second is 2306.32.

- scall ranges from 109 to 12493.

- About 75% of the scall fall under 3317.25.

d) **sread**:

Figure 4: Histplot showing count of sread



Statistical summary of the variable

```
count    8192.000000
mean      210.479980
std       198.980146
min         6.000000
25%        86.000000
50%       166.000000
75%       279.000000
max      5318.000000
Name: sread, dtype: float64
```

Observations

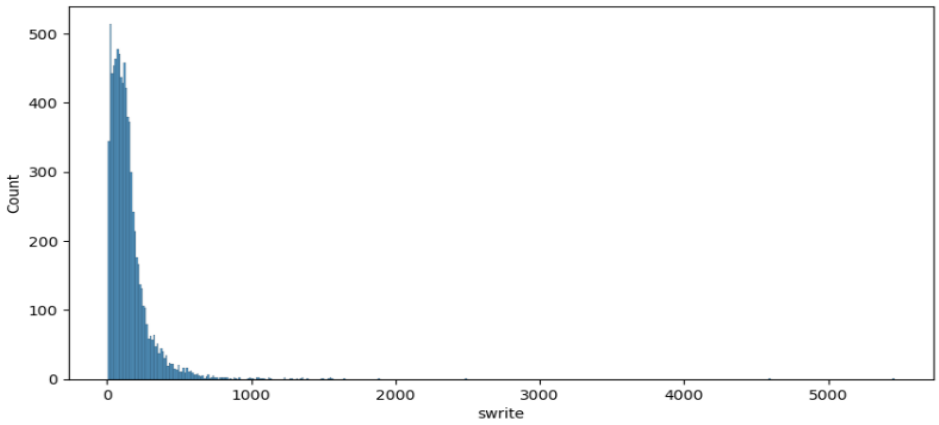
- The average number of systems read calls per second is 210.48.

- sread ranges from 6 to 5318.

- About 75% of the sread fall under 279.

e) **swrite**:

Figure 5: Histplot showing count of swrite



Statistical summary of the variable

```
count    8192.000000
mean      150.058228
std       160.478980
min         7.000000
25%        63.000000
50%       117.000000
75%       185.000000
max      5456.000000
Name: swrite, dtype: float64
```

Observations

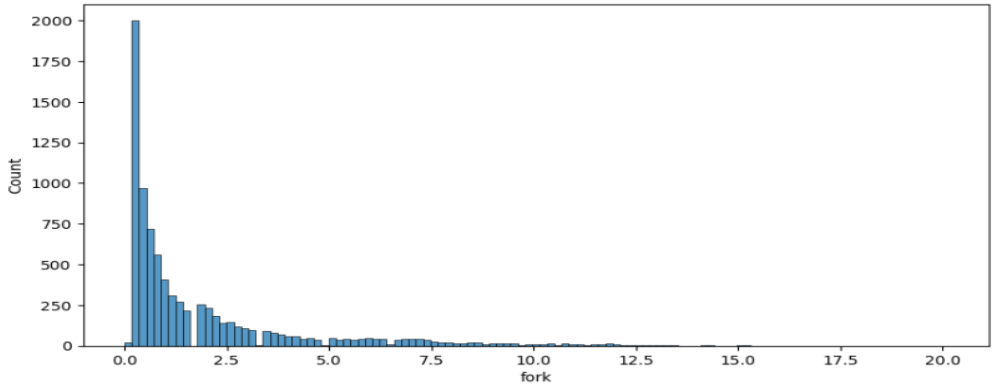
- The average number of systems write calls per second is 150.06.

- swrite ranges from 7 to 5456.

- About 25% and 75% of the swrite fall under 63 and 185.

f) **fork**:

Figure 6: Histplot showing count of fork



Statistical summary of the variable

```
count    8192.000000
mean        1.884554
std         2.479493
min         0.000000
25%         0.400000
50%         0.800000
75%         2.200000
max        20.120000
Name: fork, dtype: float64
```

Observations

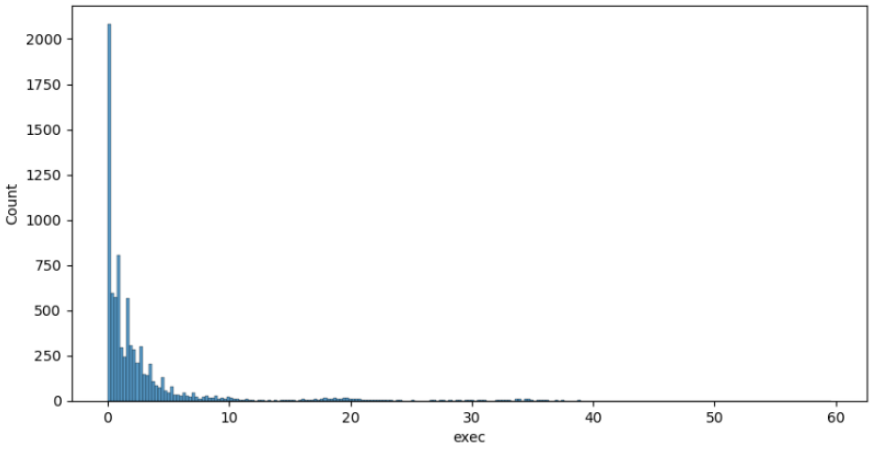
- The average number of system fork calls per second is 1.88.

- fork ranges from 0 to 20.12.

- About 25% and 75% of the fork fall under 0.40 and 2.20.

g) **exec**:

Figure 7: Histplot showing count of exec



Statistical summary of the variable

```
count    8192.000000
mean        2.791998
std         5.212456
min         0.000000
25%         0.200000
50%         1.200000
75%         2.800000
max        59.560000
Name: exec, dtype: float64
```

Observations

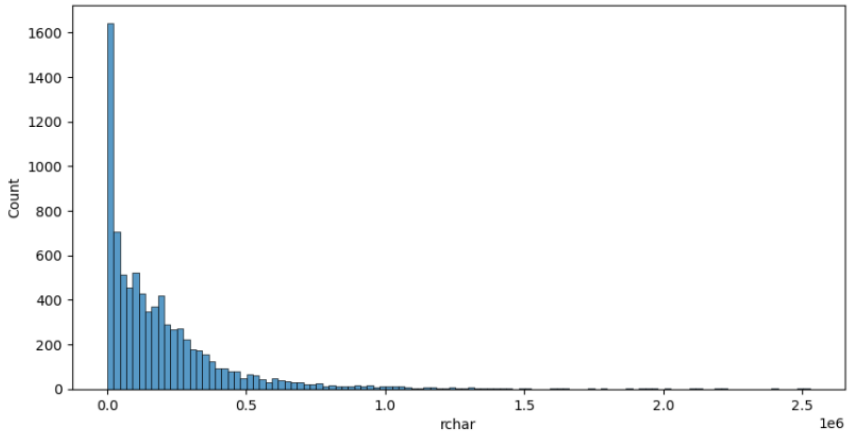
- The average number of system exec calls per second is 2.79.

- Exec ranges from 0 to 59.56.

- About 25% and 75% of the Exec fall under 0.20 and 2.80.

h) **rchar**:

Figure 8: Histplot showing count of rchar



Statistical summary of the variable

```
count    8.192000e+03
mean     1.973857e+05
std      2.383100e+05
min      2.780000e+02
25%      3.486050e+04
50%      1.278250e+05
75%      2.653948e+05
max      2.526649e+06
Name: rchar, dtype: float64
```

Observations

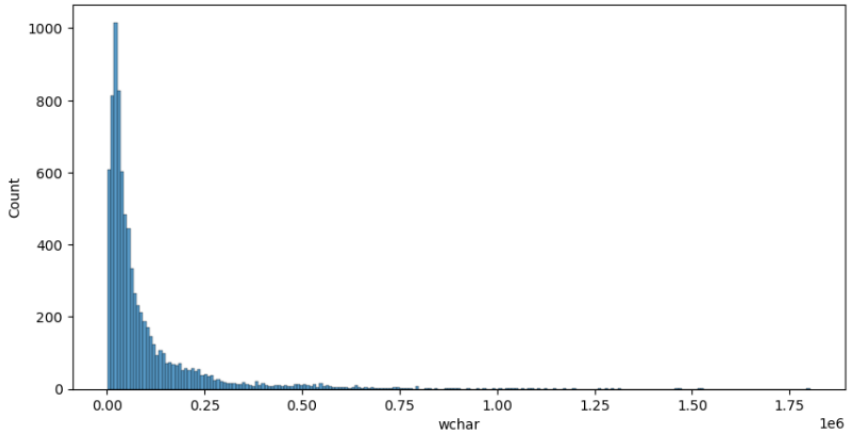
- The average number of characters transferred per second by system read calls is 1.973857e+05.

- Rchar ranges from 2.780000e+02 to 2.526649e+06.

- About 25% and 75% of the rchar fall under 3.486050e+04 and 2.653948e+05.

i) **wchar**:

Figure 9: Histplot showing count of wchar



Statistical summary of the variable

```
count    8.192000e+03
mean     9.590299e+04
std      1.407127e+05
min      1.498000e+03
25%      2.297775e+04
50%      4.665300e+04
75%      1.060370e+05
max      1.801623e+06
Name: wchar, dtype: float64
```

Observations

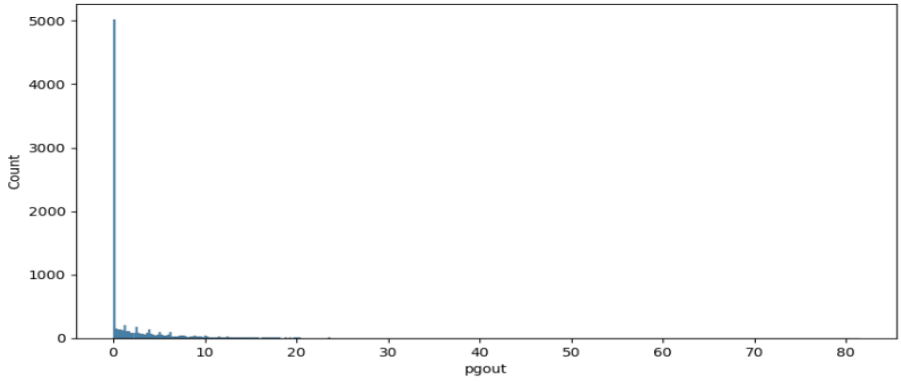
- The average number of characters transferred per second by system write calls is 9.590299e+04.

- Wchar ranges from 1.498000e+03 to 1.801623e+06.

- About 25% and 75% of the wchar fall under 2.297775e+04 and 1.060370e+05.

j) **pgout**:

Figure 10: Histplot showing count of pgout



Statistical summary of the variable

```
count    8192.000000
mean        2.285317
std         5.307038
min         0.000000
25%         0.000000
50%         0.000000
75%         2.400000
max        81.440000
Name: pgout, dtype: float64
```

Observations

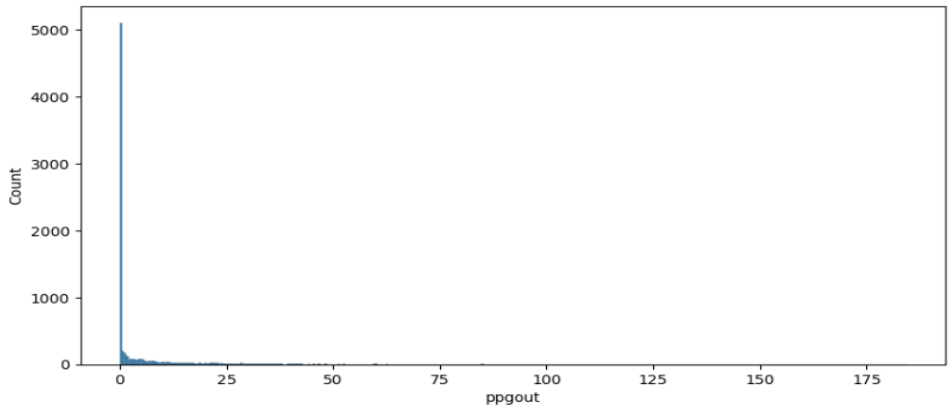
- The average number of page out requests per second is 2.29.

- pgout ranges from 0 to 81.44.

- About 75% of the pgout fall under 2.40.

k) **ppgout**:

Figure 11: Histplot showing count of ppgout



Statistical summary of the variable

```
count    8192.000000
mean        5.977229
std        15.214590
min         0.000000
25%         0.000000
50%         0.000000
75%         4.200000
max       184.200000
Name: ppgout, dtype: float64
```

Observations

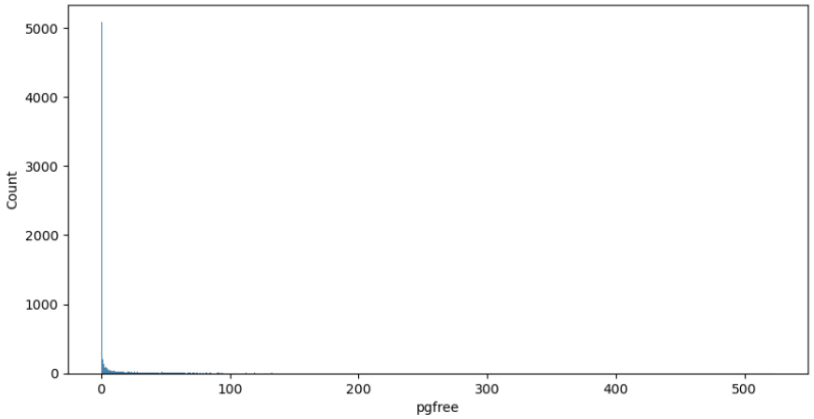
- The average number of pages, paged out per second is 5.98.

- ppgout ranges from 0 to 184.20.

- About 75% of the ppgout fall under 4.20.

l) **pgfree**:

Figure 12: Histplot showing count of pgfree



Statistical summary of the variable

```
count    8192.000000
mean       11.919712
std        32.363520
min         0.000000
25%         0.000000
50%         0.000000
75%         5.000000
max       523.000000
Name: pgfree, dtype: float64
```

Observations

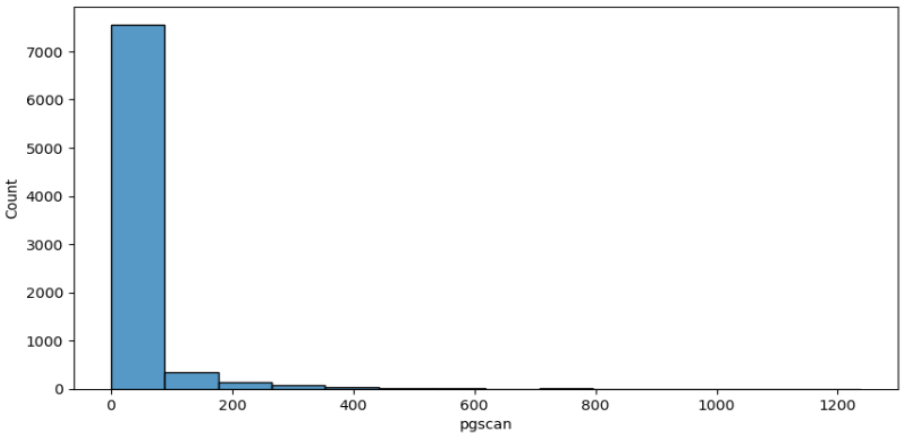
- The average number of pages per second placed on the free list is 11.92.

- pgfree ranges from 0 to 523.

- About 75% of the pgfree fall under 5.

m) **pgscan**:

Figure 13: Histplot showing count of pgscan



Statistical summary of the variable

```
count    8192.000000
mean       21.526849
std        71.141340
min         0.000000
25%         0.000000
50%         0.000000
75%         0.000000
max      1237.000000
Name: pgscan, dtype: float64
```

Observations

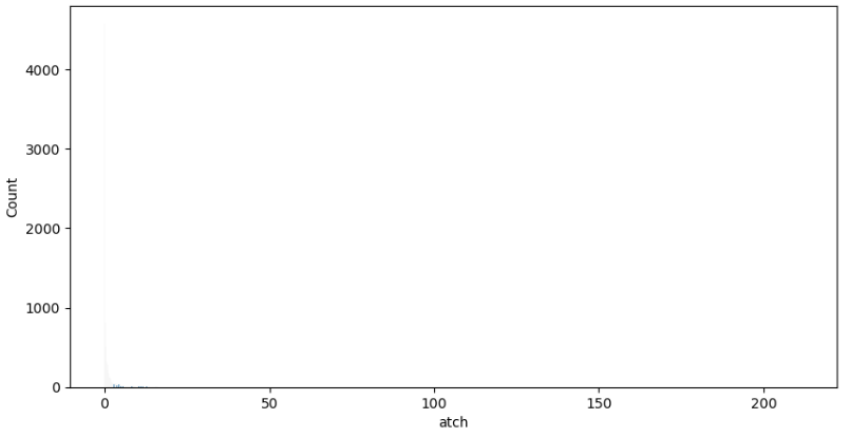
- The average number of pages checked if they can be freed per second is 21.53.

- pgscan ranges from 0 to 1237.

- About 75% of the pgscan fall under 0.

n) **atch**:

Figure 14: Histplot showing count of atch



Statistical summary of the variable

```
count    8192.000000
mean        1.127505
std         5.708347
min         0.000000
25%         0.000000
50%         0.000000
75%         0.600000
max       211.580000
Name: atch, dtype: float64
```

Observations

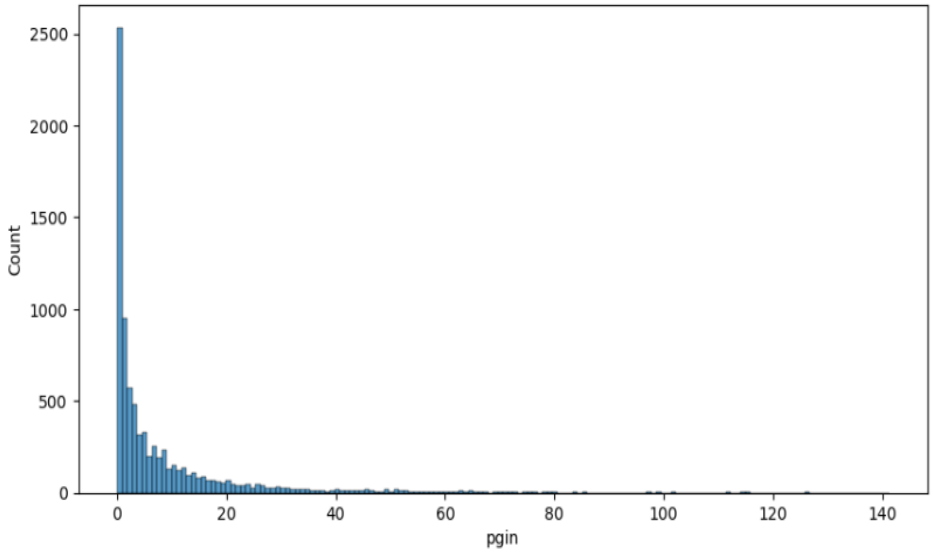
- The average number of page attaches (satisfying a page fault by reclaiming a page in memory) per second is 1.13.

- atch ranges from 0 to 211.58.

- About 25% and 75% of the atch fall under 0.00 and 0.60.

o) **pgin**:

Figure 15: Histplot showing count of pgin



Statistical summary of the variable

```
count    8192.000000
mean        8.277960
std        13.874978
min         0.000000
25%         0.600000
50%         2.800000
75%         9.765000
max       141.200000
Name: pgin, dtype: float64
```

Observations

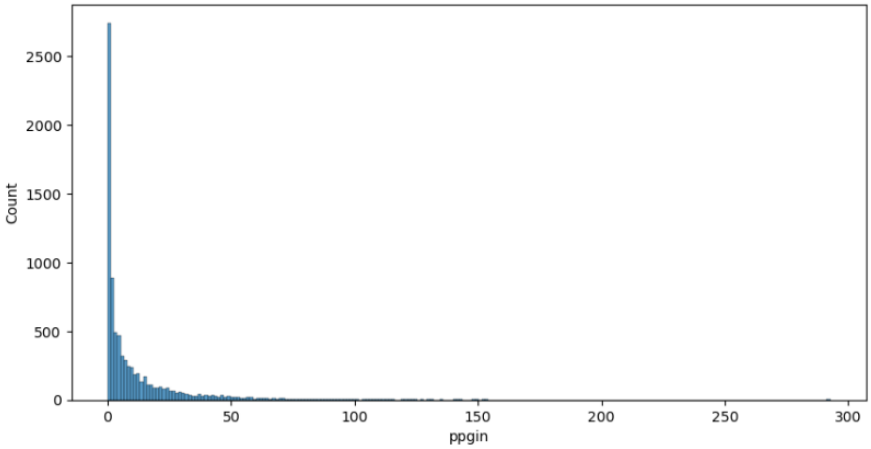
- The average number of page-in requests per second is 8.28.

- pgin ranges from 0 to 141.20.

- About 75% of the pgin fall under 9.77.

p) **ppgin**:

Figure 16: Histplot showing count of ppgin



Statistical summary of the variable

```
count    8192.000000
mean       12.388586
std        22.281318
min         0.000000
25%         0.600000
50%         3.800000
75%        13.800000
max       292.610000
Name: ppgin, dtype: float64
```

Observations

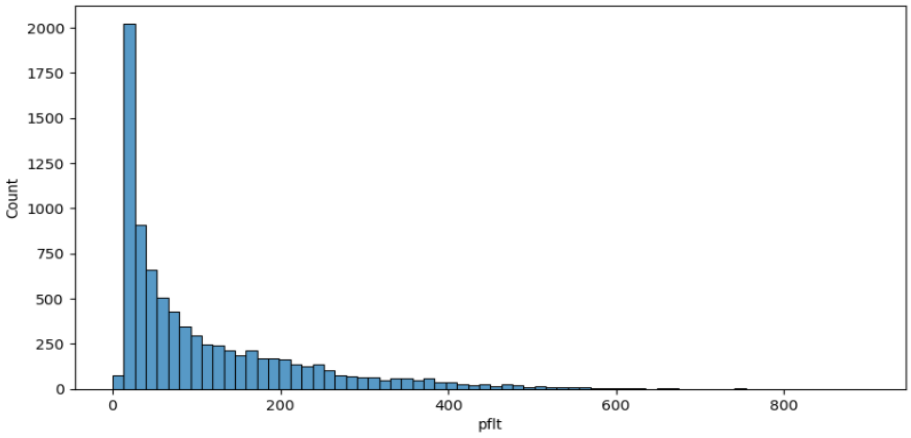
- The average number of pages paged in per second is 12.39.

- ppgin ranges from 0 to 13.80.

- About 75% of the ppgin fall under 13.80.

q) **pflt**:

Figure 17: Histplot showing count of pflt



Statistical summary of the variable

```
count    8192.000000
mean      109.793799
std       114.419221
min         0.000000
25%        25.000000
50%        63.800000
75%       159.600000
max       899.800000
Name: pflt, dtype: float64
```

Observations

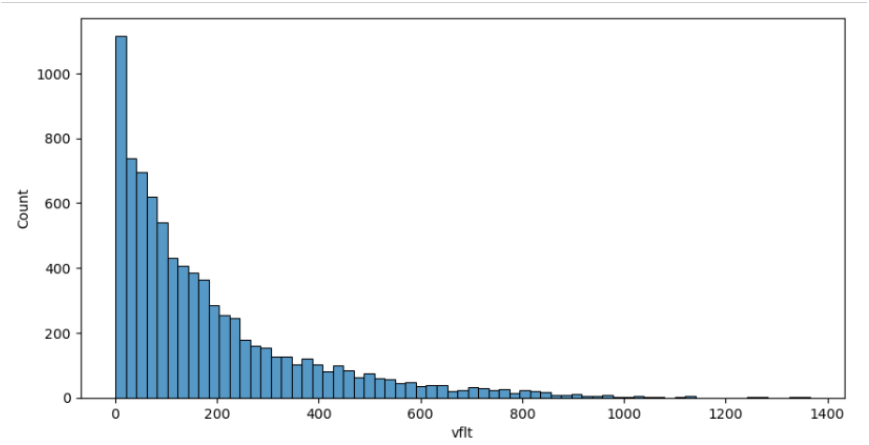
- The average number of page faults caused by protection errors (copy-on-writes) is 109.79.

- pflt ranges from 0 to 899.80.

- About 75% of the pflt fall under 159.60.

r) **vflt**:

Figure 18: Histplot showing count of vflt



Statistical summary of the variable

```
count    8192.000000
mean      185.315796
std       191.000603
min         0.200000
25%        45.400000
50%       120.400000
75%       251.800000
max      1365.000000
Name: vflt, dtype: float64
```

Observations

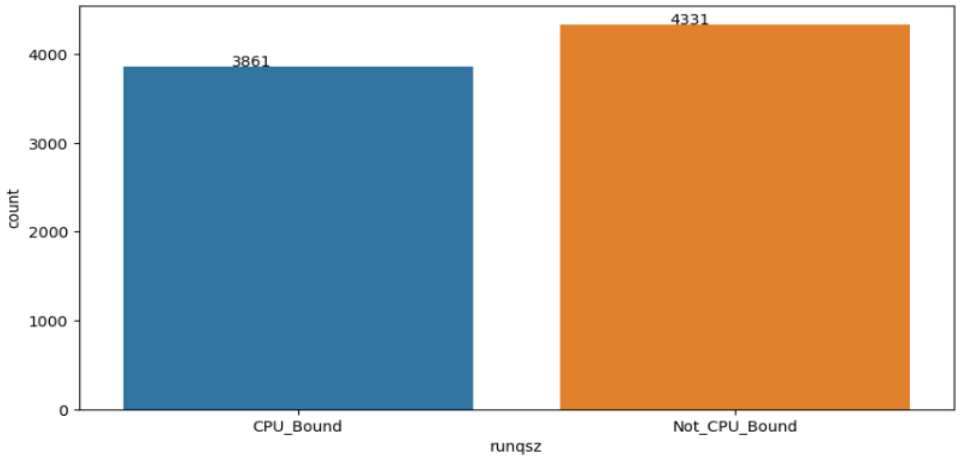
- The average number of page faults caused by address translation is 185.32.

- vflt ranges from 0.20 to 1365.

- About 75% of the vflt fall under 251.80.

s) **runqsz**:

Figure 19: Countplot showing count of runqsz



Statistical summary of the variable

```
count              8192
unique                2
top       Not_CPU_Bound
freq               4331
Name: runqsz, dtype: object
```

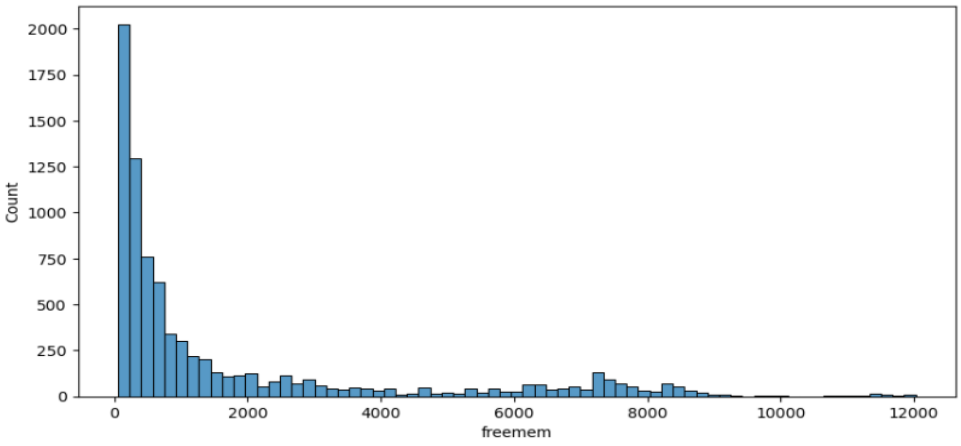
Observations

- There are 2 unique values in runqsz variable.

- About 4331 of 8192 are Not_CPU_Bound.

t) **freemem**:

Figure 20: Histplot showing count of freemem



Statistical summary of the variable

```
count     8192.000000
mean      1763.456299
std       2482.104511
min         55.000000
25%        231.000000
50%        579.000000
75%       2002.250000
max      12027.000000
Name: freemem, dtype: float64
```

Observations

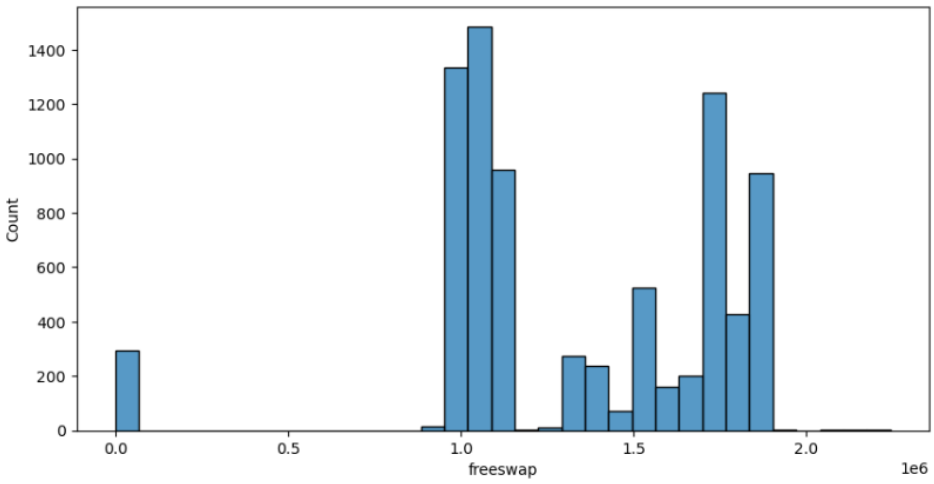
- The average number of memory pages available to user processes is 1763.46.

- freemem ranges from 55 to 12027.

- About 75% of the freemem fall under 2002.25.

u) **freeswap**:

Figure 21: Histplot showing count of freeswap



Statistical summary of the variable

```
count    8.192000e+03
mean     1.328126e+06
std      4.220194e+05
min      2.000000e+00
25%      1.042624e+06
50%      1.289290e+06
75%      1.730380e+06
max      2.243187e+06
Name: freeswap, dtype: float64
```

Observations

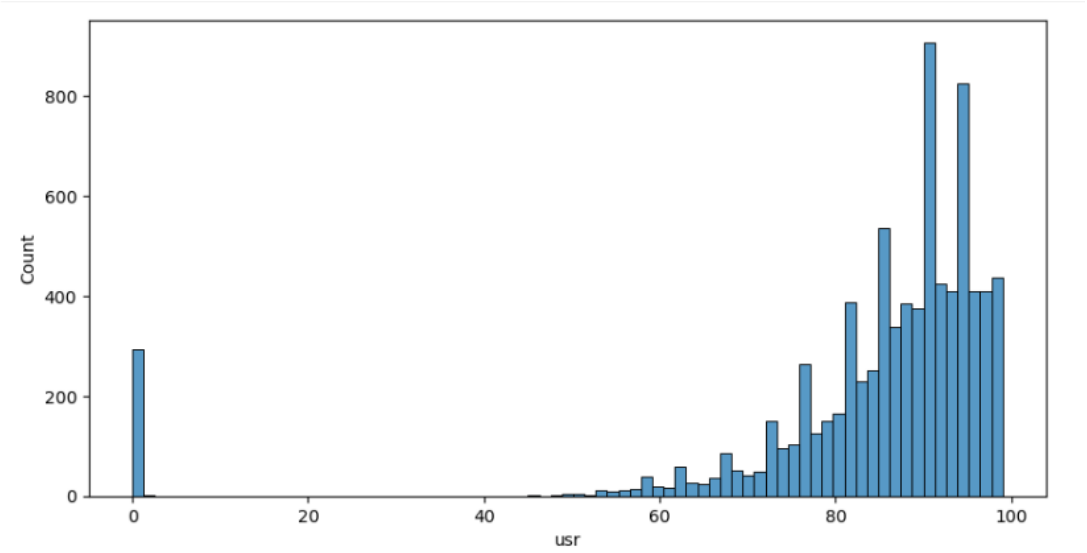
- The average number of disk blocks available for page swapping is 1.328126e+06.

- freeswap ranges from 2.000000e+00 to 2.243187e+06.

- About 75% of the freeswap fall under 1.730380e+06.

v) **usr**:

Figure 22: Histplot showing count of usr



Statistical summary of the variable

```
count    8192.000000
mean       83.968872
std        18.401905
min         0.000000
25%        81.000000
50%        89.000000
75%        94.000000
max        99.000000
Name: usr, dtype: float64
```

Observations

- The average Portion of time (%) that cpus run in user mode is 83.97.

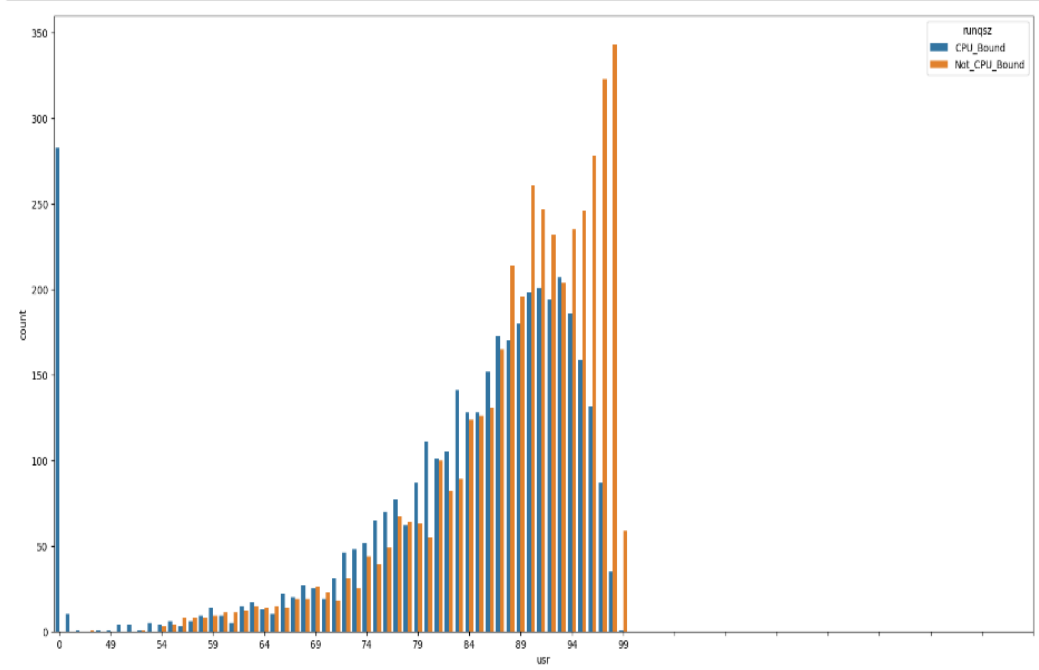
- usr ranges from 0 to 99.

- About 75% of the usr fall under 94.

- **<u>Bivariate Analysis and Multivariate Analysis:</u>**

a) runqsz and usr:

Figure 23: Histplot showing count of runqsz and usr

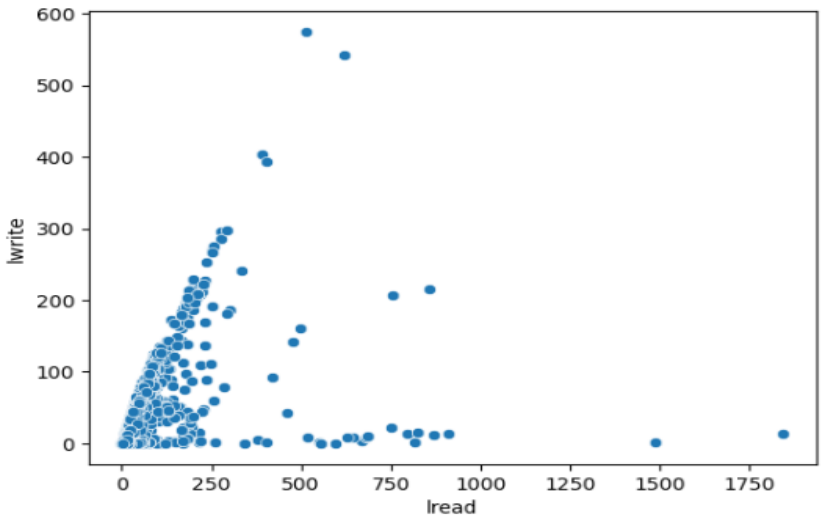


Observations

- As the portion of time (%) that cpus run in user mode increases, CPU bound decreases.

b) lread and lwrite:

Figure 24: Scatterplot showing number of lread and lread



	lread	lwrite
lread	1.000000	0.533737
lwrite	0.533737	1.000000

	count	mean	std	min	25%	50%	75%	max
lread	8192.0	19.559692	53.353799	0.0	2.0	7.0	20.0	1845.0
lwrite	8192.0	13.106201	29.891726	0.0	0.0	1.0	10.0	575.0

Observations

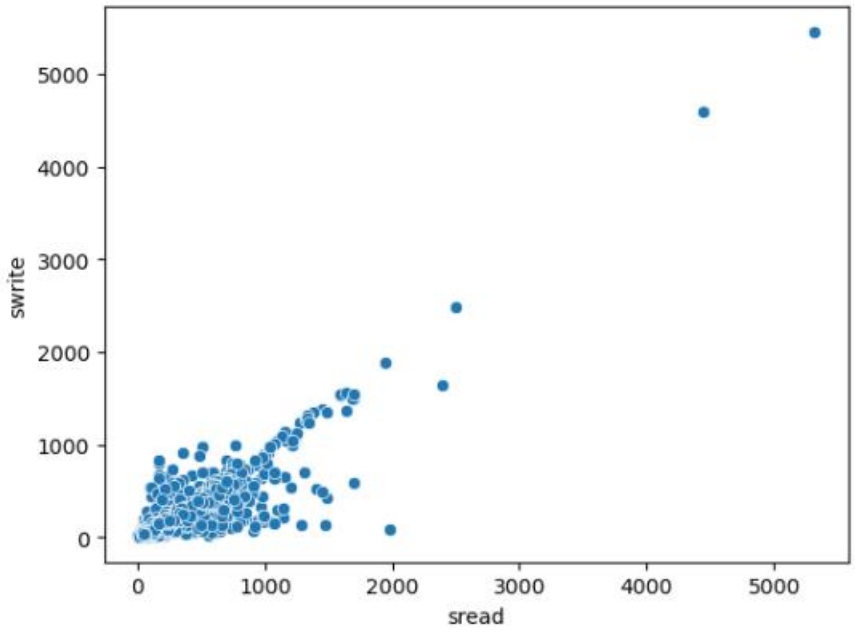
- The above graph states that there is a positive correlation of 0.53 between the lread and lwrite.

- The mean of lread and lwrite is 19.56 and 13.11.

- The std of lread and lwrite is 53.35 and 29.89.

c) sread and swrite:

Figure 25: Scatterplot showing number of sread and swrite



	sread	swrite
sread	1.000000	0.881069
swrite	0.881069	1.000000

	count	mean	std	min	25%	50%	75%	max
sread	8192.0	210.479980	198.980146	6.0	86.0	166.0	279.0	5318.0
swrite	8192.0	150.058228	160.478980	7.0	63.0	117.0	185.0	5456.0

Observations

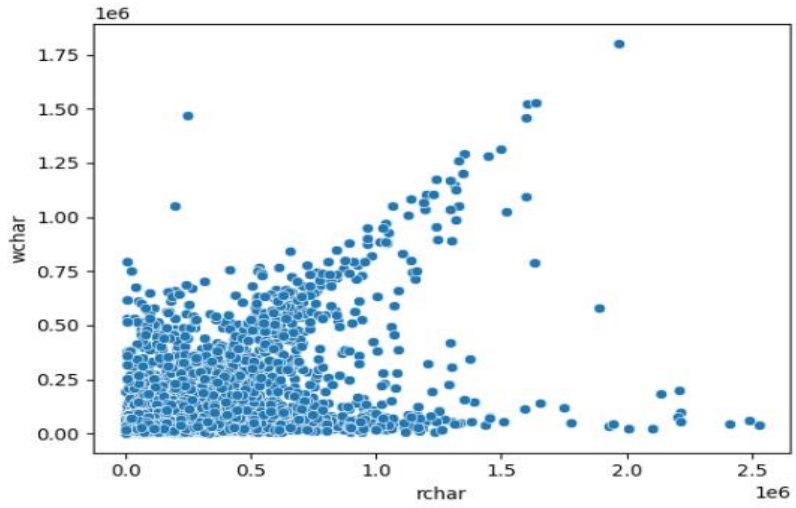
- The above graph states that there is a positive correlation of 0.88 between the sread and swrite.

- The mean of sread and swrite is 210.48 and 150.06.

- The std of sread and swrite is 198.98 and 160.48.

d) rchar and wchar:

Figure 26: Scatterplot showing number of rchar and wchar



	rchar	wchar
rchar	1.000000	0.499825
wchar	0.499825	1.000000

	count	mean	std	min	25%	50%	75%	max
rchar	8192.0	197385.728363	238310.037735	278.0	34860.50	127825.0	265394.75	2526649.0
wchar	8192.0	95902.992785	140712.688639	1498.0	22977.75	46653.0	106037.00	1801623.0

Observations

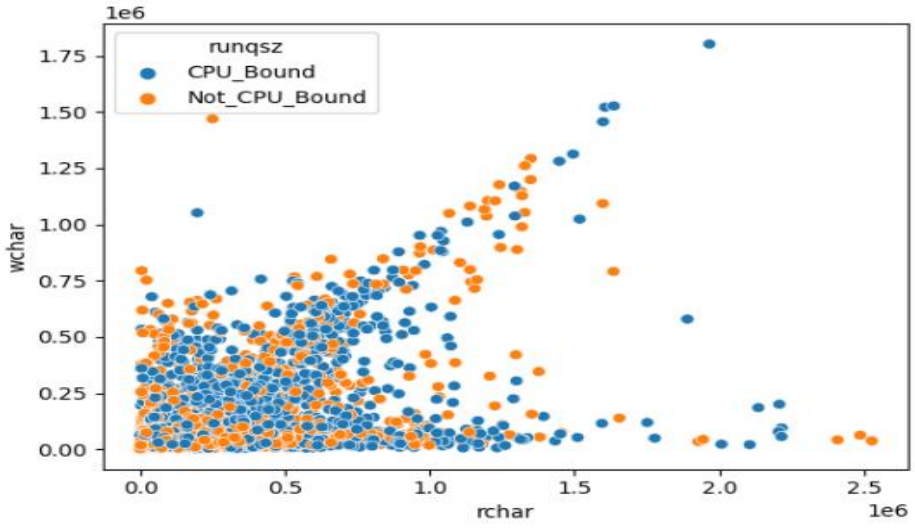
- The above graph states that there is a positive correlation of 0.50 between the rchar and wchar.

- The mean of rchar and wchar is 197385.73 and 95903.

- The std of rchar and wchar is 238310.04 and 140712.69.

e) rchar, wchar and runqsz:

Figure 27: Scatterplot showing number of rchar, wchar and runqsz



	rchar	wchar
rchar	1.000000	0.499825
wchar	0.499825	1.000000

```
count              8192
unique                2
top       Not_CPU_Bound
freq               4331
Name: runqsz, dtype: object
```

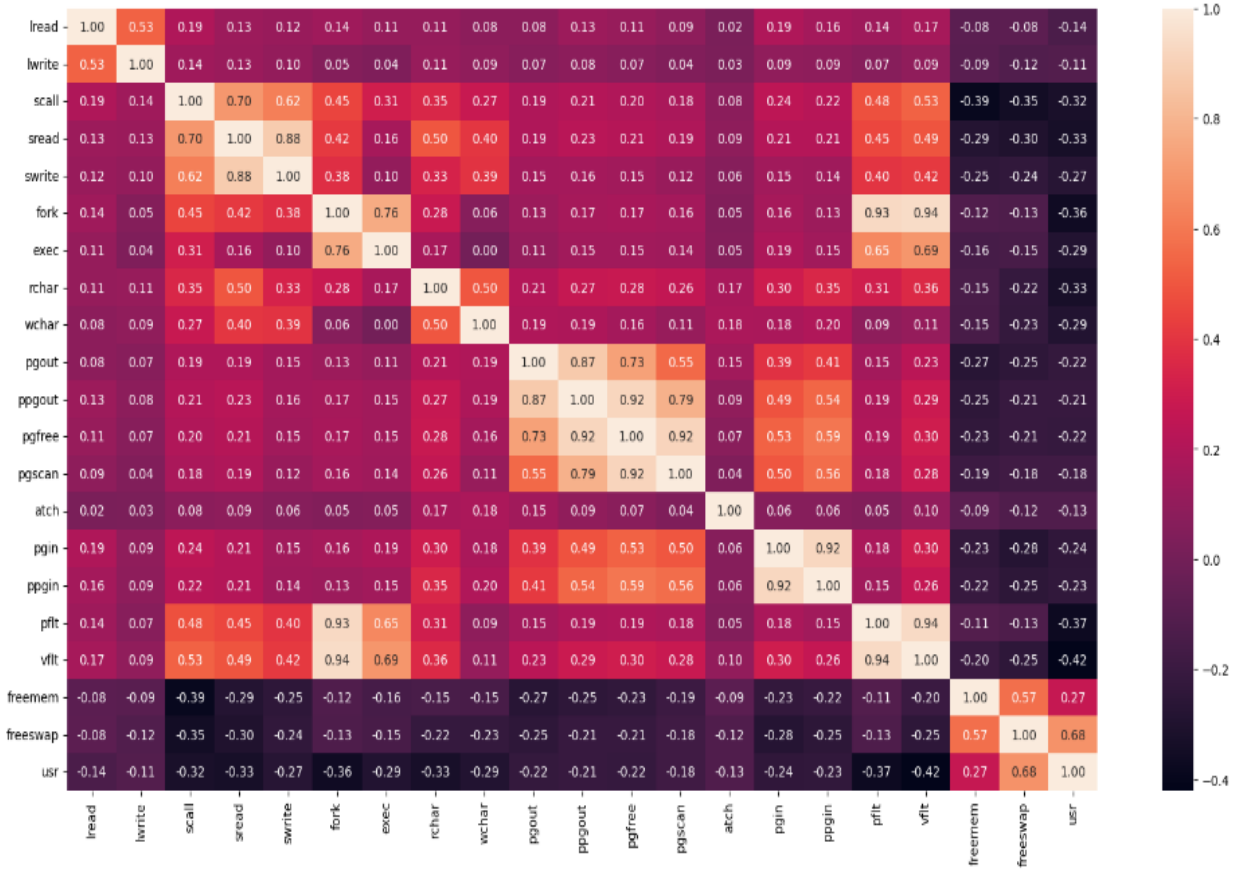
Observations

- The above graph states that there is a positive correlation of 0.50 between the rchar and wchar.

- About 4331 of 8192 are Not_CPU_Bound.

f) Correlation between the variables:

Figure 28: Heatmap showing correlation between variables



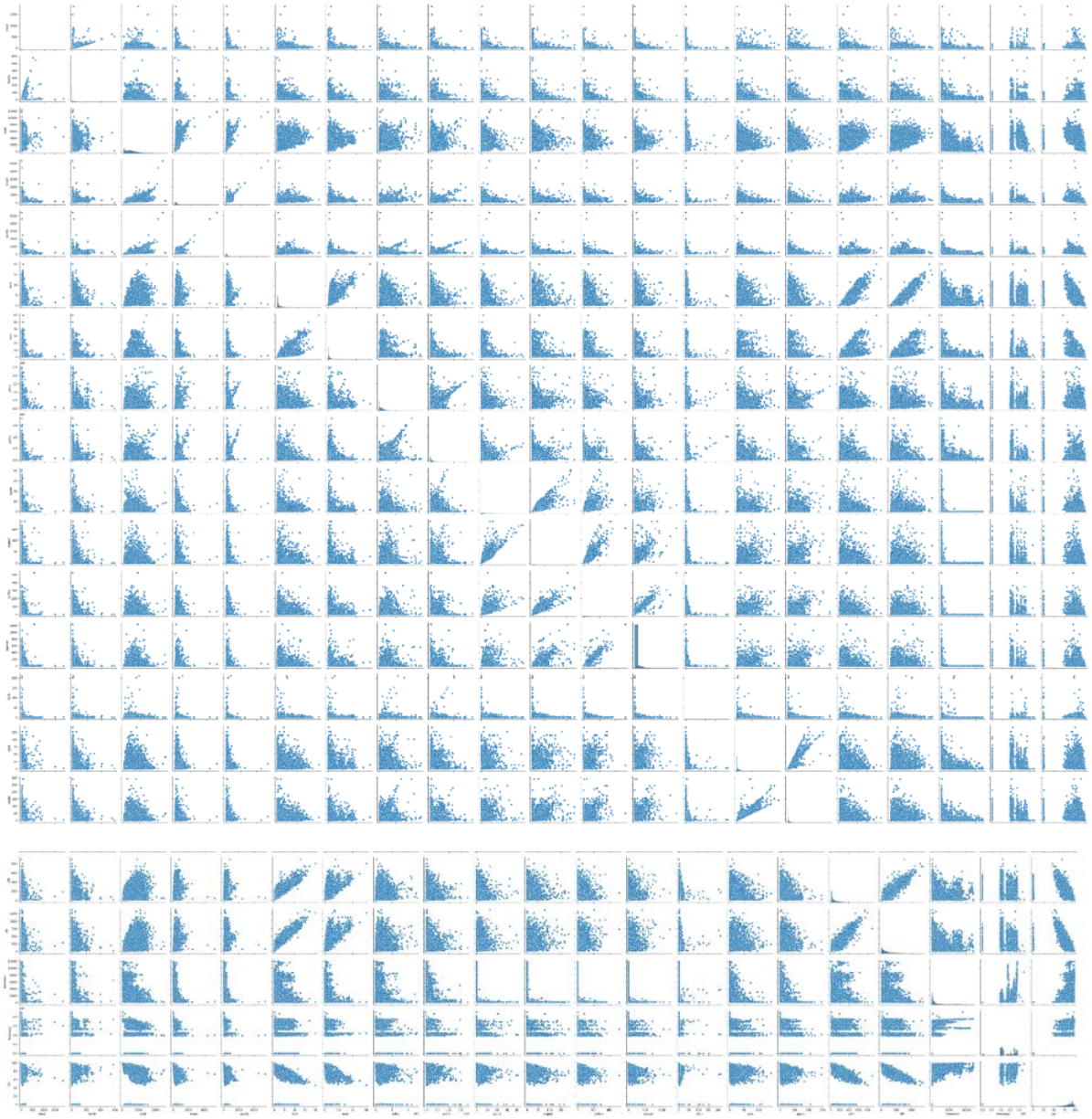
Observations

- There are few variables which are highly correlated to each other, as their values are close to 1.

- Few variables have negative values which represents negative correlation between each other.

g) Pairplot:

Figure 29: Pairplot of the given variables



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**.**

Ans:

- **Null values:** There are 104 missing values in rchar column and 15 missing values in wchar column.

```
rchar    104
wchar     15
dtype: int64
```

Imputing null values: We can impute the null values with mean or median for numeric variables and for categorical variables, mode can be used. However, we are proceeding the imputation for missing values with Mean.

Table 6: Table showing null values after mean imputation

```
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
```

- **Values which are equal to zero:**

  Number of zeroes in each variable are as follows:

Table 7: Table showing number of zeroes in each variable

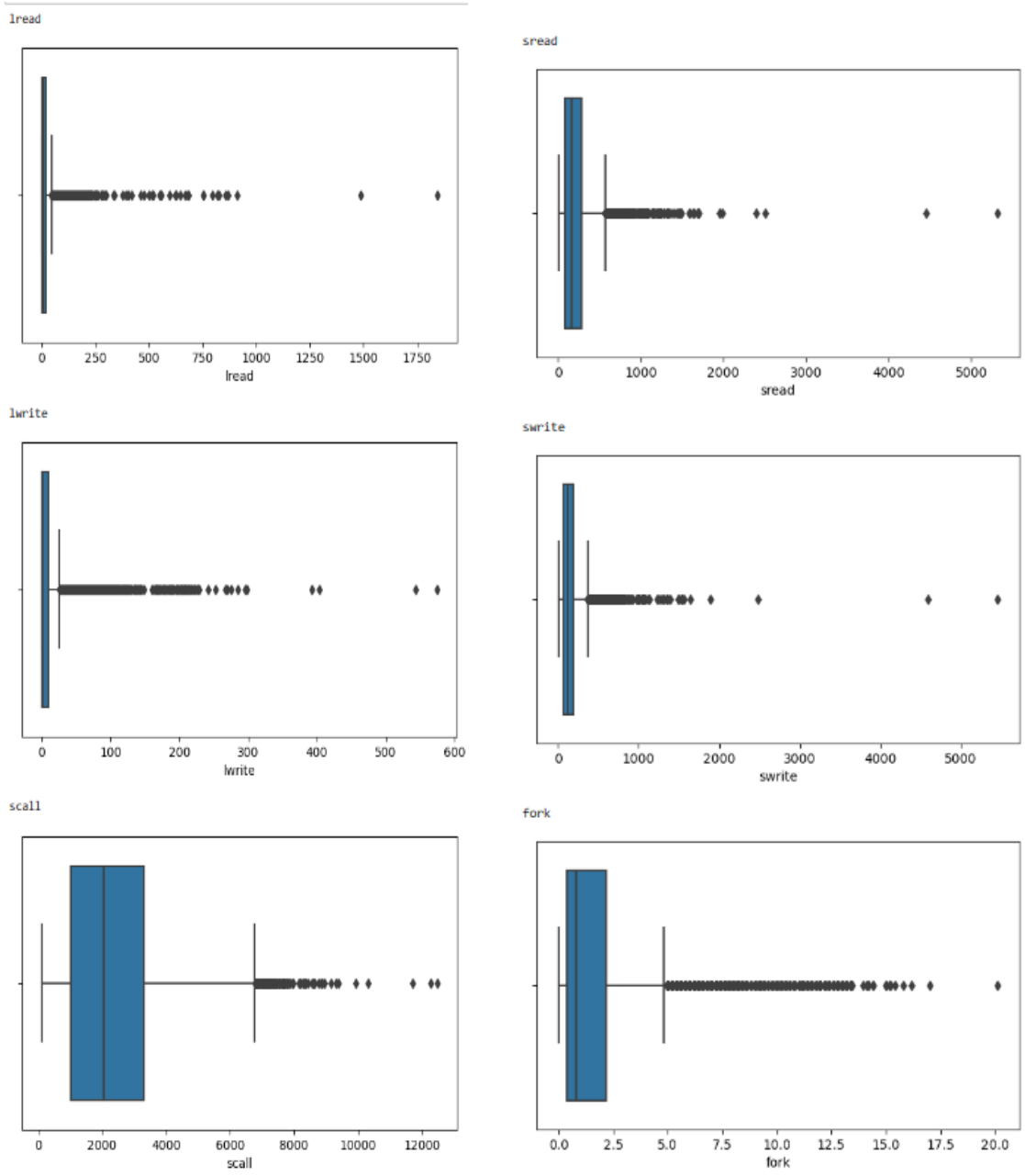
```
lread 675
lwrite 2684
scall 0
sread 0
swrite 0
fork 21
exec 21
rchar 0
wchar 0
pgout 4878
ppgout 4878
pgfree 4869
pgscan 6448
atch 4575
pgin 1220
ppgin 1220
pflt 3
vflt 0
runqsz 0
freemem 0
freeswap 0
usr 283
```

We are not changing or dropping zeroes, as few of the values can be actually zeroes and changing the same would lead to wrong interpretation of the data. However, in this case the best practice would be to return back to the business for more clarity on the data or for more data.

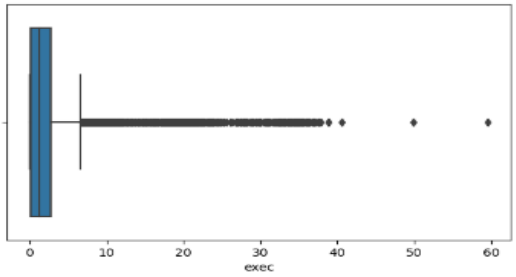
- **Outliers**:

  We can note that, there are outliers for the given variables which can interpreted with the help of boxplots.

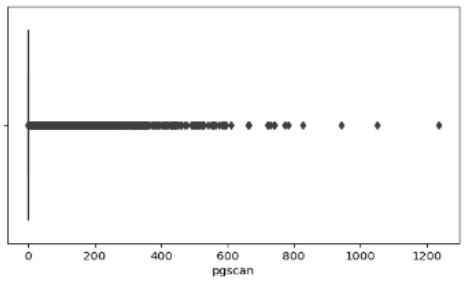
Figure 30: Boxplots before treating outliers



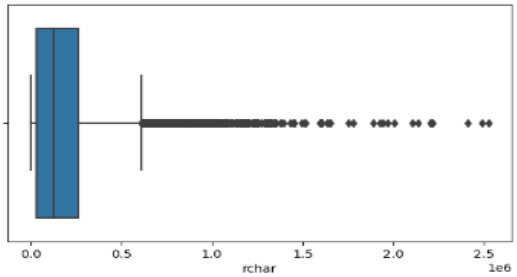
exec



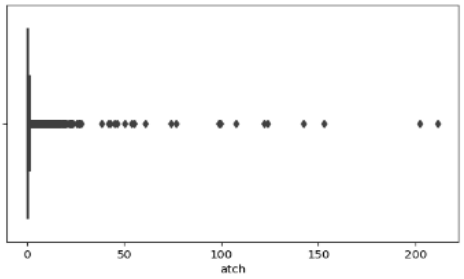
pgscan



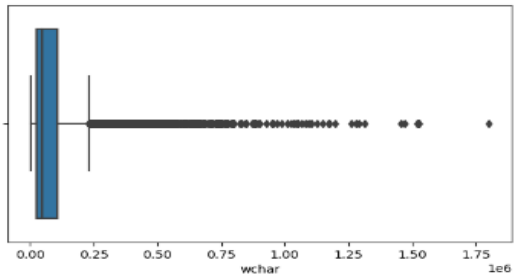
rchar



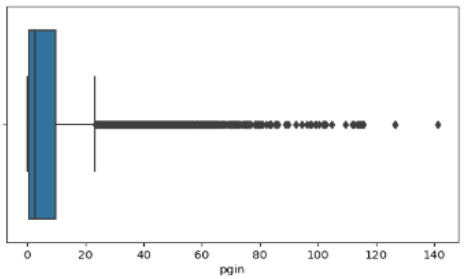
atch



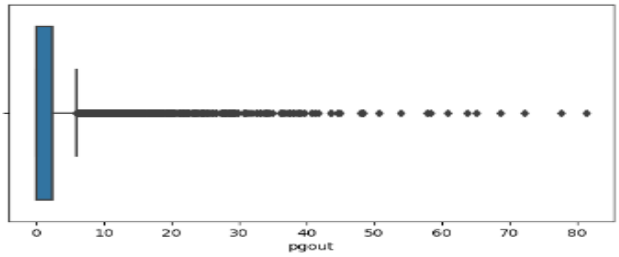
wchar



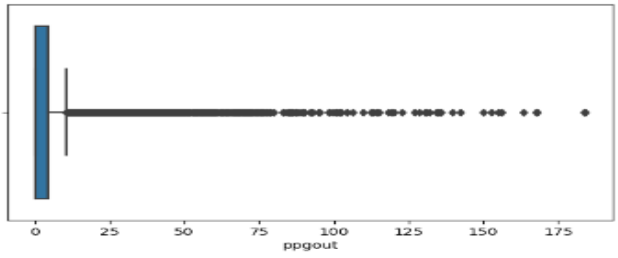
pgin



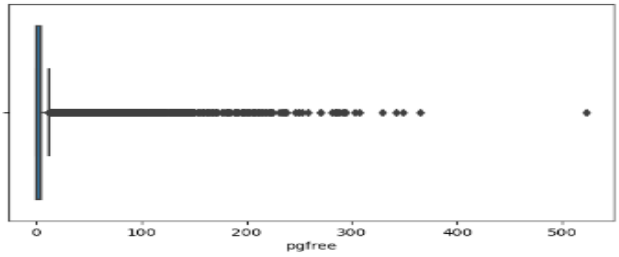
pgout

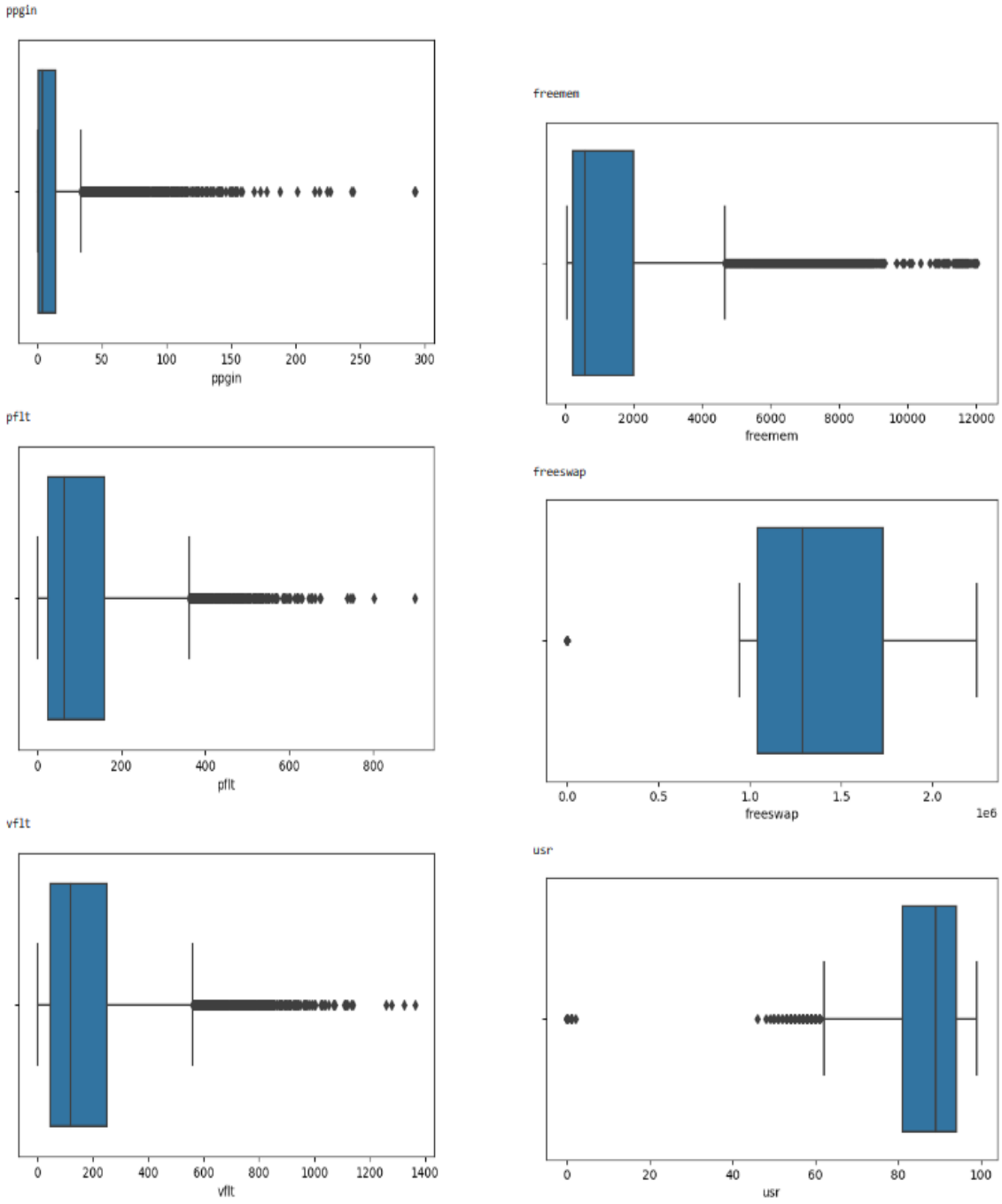


ppgout



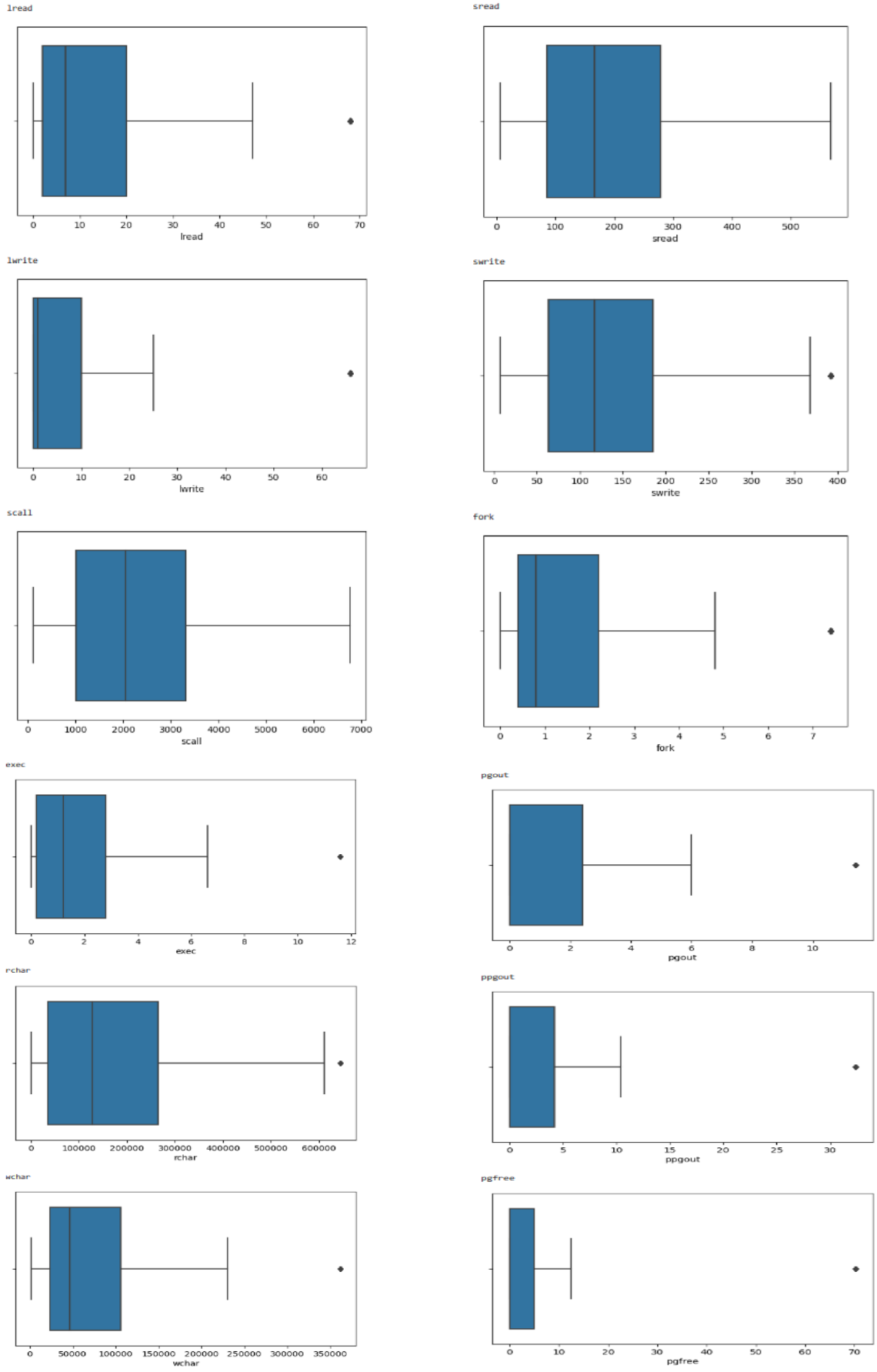
pgfree



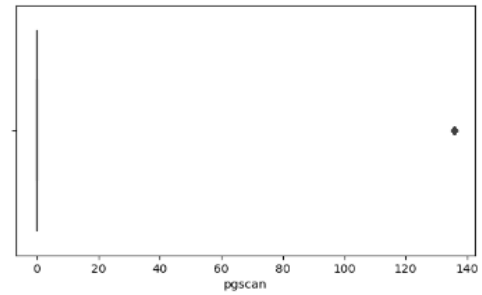


- **Treating outliers**: We are treating outliers using the IQR method i.e., by using 95th percentile method in order to capture the maximum data.

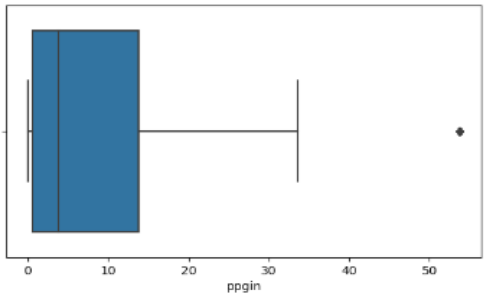
Figure 31: Boxplots after treating outliers



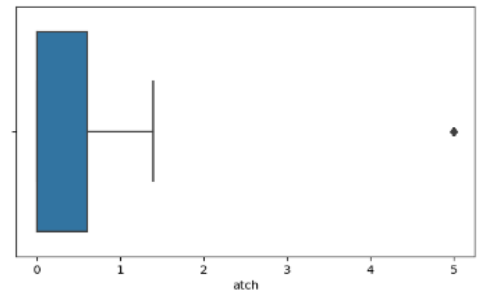
pgscan



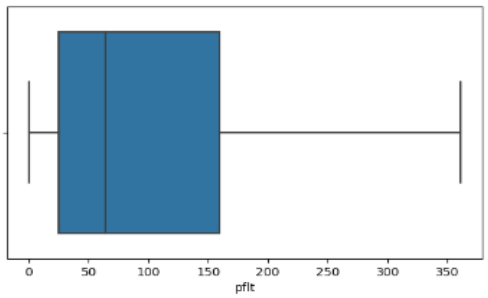
ppgin



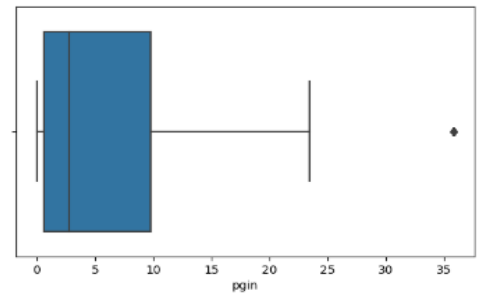
atch



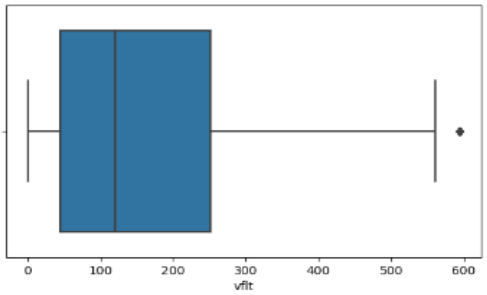
pflt



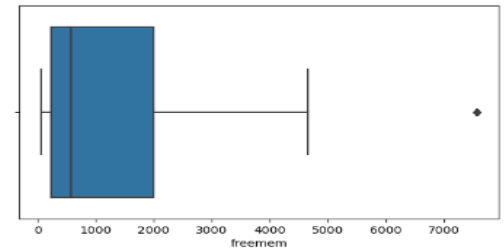
pgin



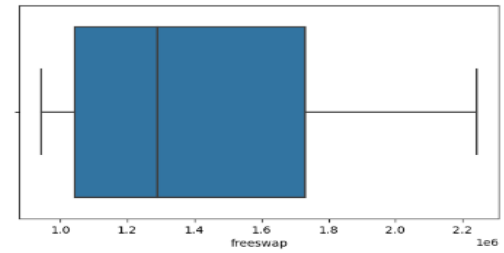
vflt



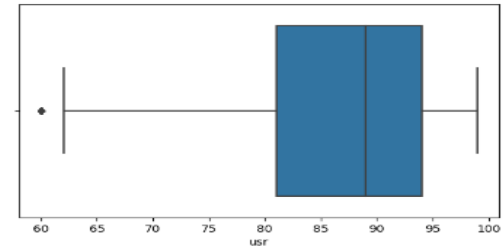
freemem



freeswap



usr



- **Duplicate values:** There are no duplicate values in the given dataset.

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.

Ans:

- Converting the 'object' type variable to dummy variable.
- Assigning all independent variables to 'X'.
- Assigning dependent variables to 'Y'.
- Splitting the data into train and test at 70:30 ratio.
- Please note, we are not scaling the data.

a) Using Linear Model from Sci-kit learn library:

- Initializing and Fitting the **Linear Regression Model.**
- Linear Regression Model **coefficients:**

Table 8: Table showing coefficients of Linear Regression Model

```
array([[-5.76488445e-02,  2.49030219e-02, -4.54185741e-04,
        -1.61775223e-04, -7.65745511e-03, -1.03702216e-01,
        -3.19498434e-01, -5.30957395e-06, -5.31729461e-06,
        -2.88847083e-01,  2.96657543e-02,  1.23625141e-02,
        -3.63100470e-03,  5.67706633e-02, -8.42037100e-02,
         6.61712293e-03, -1.69209099e-02, -1.28464412e-02,
         1.01279536e-04,  4.27556288e-06,  2.13459646e+00]])

The coefficient for lread is -0.05764884448399361
The coefficient for lwrite is 0.02490302190330532
The coefficient for scall is -0.00045418574139886566
The coefficient for sread is -0.0001617752227823815
The coefficient for swrite is -0.00765745510568917
The coefficient for fork is -0.10370221638050504
The coefficient for exec is -0.3194984336517109
The coefficient for rchar is -5.309573954307643e-06
The coefficient for wchar is -5.317294608399103e-06
The coefficient for pgout is -0.28884708330302356
The coefficient for ppgout is 0.029665754279059237
The coefficient for pgfree is 0.012362514057164806
The coefficient for pgscan is -0.0036310046981802933
The coefficient for atch is 0.056770663286474725
The coefficient for pgin is -0.08420371002742094
The coefficient for ppgin is 0.006617122926777838
The coefficient for pflt is -0.01692090987984253
The coefficient for vflt is -0.01284644117719284
The coefficient for freemem is 0.0001012795362156975
The coefficient for freeswap is 4.275562878417139e-06
The coefficient for runqsz_Not_CPU_Bound is 2.1345964552279133
```

- Linear Regression Model **intercept:**

  The intercept of model is **88.9830.**

```
array([88.98300559])
```

- **Predicted Train data:**

```
array([[93.19355319],
       [60.97935545],
       [92.2965406 ],
       ...,
       [94.147894  ],
       [87.6946093 ],
       [98.8729789 ]])
```

- **R square on Training and Testing data:**

  R square on **Training** data is **0.6845**

  R square on **Testing** data is **0.7243**

- **RMSE on Training and Testing data:**

  RMSE on **Training** data is **5.5896**

  RMSE on **Testing** data is **5.1181**

b) Linear Model using statsmodel (OLS):

- Initializing and Fitting the **Linear Regression Model.**
- OLS Regression results:

Table 9: Table showing OLS Regression results on training data

OLS Regression Results

Dep. Variable:	usr	R-squared:	0.690
Model:	OLS	Adj. R-squared:	0.689
Method:	Least Squares	F-statistic:	605.1
Date:	Wed, 06 Sep 2023	Prob (F-statistic):	0.00
Time:	09:10:52	Log-Likelihood:	-18004.
No. Observations:	5734	AIC:	3.605e+04
Df Residuals:	5712	BIC:	3.620e+04
Df Model:	21		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	88.9830	0.522	170.454	0.000	87.960	90.006
lread	-0.0576	0.008	-7.553	0.000	-0.073	-0.043
lwrite	0.0249	0.006	4.231	0.000	0.013	0.036
scall	-0.0005	8.38e-05	-5.417	0.000	-0.001	-0.000
sread	-0.0002	0.001	-0.120	0.904	-0.003	0.002
swrite	-0.0077	0.002	-4.404	0.000	-0.011	-0.004
fork	-0.1037	0.106	-0.974	0.330	-0.312	0.105
exec	-0.3195	0.038	-8.348	0.000	-0.395	-0.244
rchar	-5.31e-06	6.05e-07	-8.772	0.000	-6.5e-06	-4.12e-06
wchar	-5.317e-06	8.96e-07	-5.938	0.000	-7.07e-06	-3.56e-06
pgout	-0.2888	0.044	-6.525	0.000	-0.376	-0.202
ppgout	0.0297	0.019	1.533	0.125	-0.008	0.068
pgfree	0.0124	0.008	1.505	0.132	-0.004	0.028
pgscan	-0.0036	0.003	-1.411	0.158	-0.009	0.001
atch	0.0568	0.052	1.093	0.275	-0.045	0.159
pgin	-0.0842	0.022	-3.899	0.000	-0.127	-0.042
ppgin	0.0066	0.015	0.455	0.649	-0.022	0.035
pflt	-0.0169	0.002	-7.186	0.000	-0.022	-0.012
vflt	-0.0128	0.002	-7.339	0.000	-0.016	-0.009
freemem	0.0001	4.13e-05	2.451	0.014	2.03e-05	0.000
freeswap	4.276e-06	3.23e-07	13.254	0.000	3.64e-06	4.91e-06
runqsz_Not_CPU_Bound	2.1346	0.163	13.120	0.000	1.816	2.454

Omnibus:	3233.335	Durbin-Watson:	2.028
Prob(Omnibus):	0.000	Jarque-Bera (JB):	27922.211
Skew:	-2.608	Prob(JB):	0.00
Kurtosis:	12.470	Cond. No.	1.01e+07

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 1.01e+07. This might indicate that there are
strong multicollinearity or other numerical problems.

Table 10: Table showing OLS Regression results on testing data

OLS Regression Results

Dep. Variable:	usr	R-squared:	0.724
Model:	OLS	Adj. R-squared:	0.722
Method:	Least Squares	F-statistic:	304.7
Date:	Wed, 06 Sep 2023	Prob (F-statistic):	0.00
Time:	10:44:01	Log-Likelihood:	-7501.1
No. Observations:	2458	AIC:	1.505e+04
Df Residuals:	2436	BIC:	1.517e+04
Df Model:	21		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	90.9361	0.746	121.867	0.000	89.473	92.399
lread	-0.0548	0.010	-5.294	0.000	-0.075	-0.034
lwrite	0.0166	0.008	2.079	0.038	0.001	0.032
scall	-0.0004	0.000	-3.297	0.001	-0.001	-0.000
sread	-0.0046	0.002	-2.533	0.011	-0.008	-0.001
swrite	-0.0057	0.002	-2.316	0.021	-0.010	-0.001
fork	-0.0369	0.144	-0.257	0.797	-0.319	0.245
exec	-0.4144	0.054	-7.710	0.000	-0.520	-0.309
rchar	-4.792e-06	8.25e-07	-5.807	0.000	-6.41e-06	-3.17e-06
wchar	-8.3e-06	1.25e-06	-6.652	0.000	-1.07e-05	-5.85e-06
pgout	-0.2013	0.064	-3.143	0.002	-0.327	-0.076
ppgout	-0.0126	0.028	-0.456	0.648	-0.067	0.042
pgfree	0.0152	0.011	1.350	0.177	-0.007	0.037
pgscan	-0.0019	0.004	-0.519	0.603	-0.009	0.005
atch	0.0647	0.074	0.869	0.385	-0.081	0.211
pgin	0.0059	0.028	0.210	0.834	-0.049	0.061
ppgin	-0.0454	0.018	-2.492	0.013	-0.081	-0.010
pflt	-0.0221	0.004	-6.255	0.000	-0.029	-0.015
vflt	-0.0088	0.002	-3.665	0.000	-0.014	-0.004
freemem	0.0001	5.61e-05	2.098	0.036	7.7e-06	0.000
freeswap	3.541e-06	4.53e-07	7.819	0.000	2.65e-06	4.43e-06
runqsz_Not_CPU_Bound	1.5154	0.231	6.573	0.000	1.063	1.967

Omnibus:	1520.440	Durbin-Watson:	1.971
Prob(Omnibus):	0.000	Jarque-Bera (JB):	16945.632
Skew:	-2.798	Prob(JB):	0.00
Kurtosis:	14.582	Cond. No.	1.03e+07

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 1.03e+07. This might indicate that there are
strong multicollinearity or other numerical problems.

- **R square on Training and Testing data:**

  R square on **Training** data is **0.690**

  R square on **Testing** data is **0.724**

- **Adjusted R-squared value** on **Training** data **is 0.689.**
- **Adjusted R-squared value** on **Testing** data **is 0.722.**

- **RMSE on Training and Testing data:**

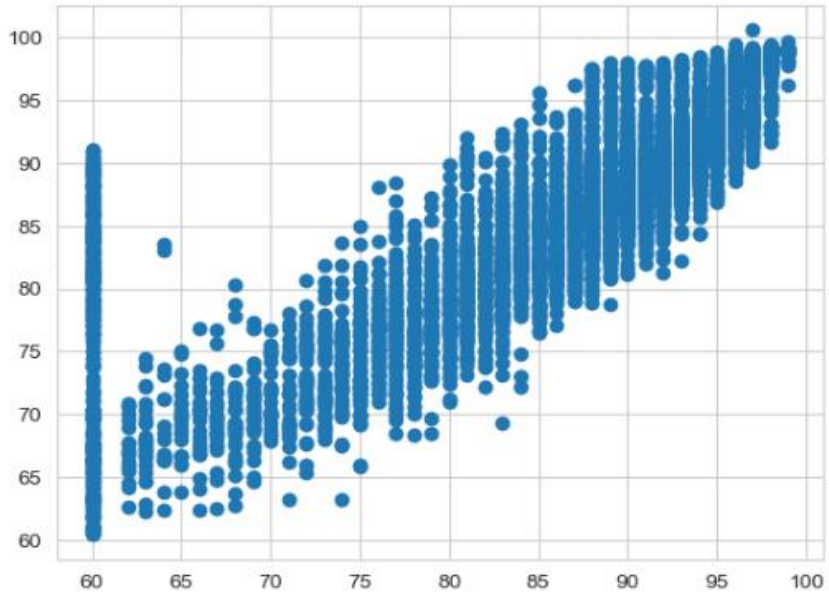
  RMSE on **Training** data is **5.5896**

  RMSE on **Testing** data is **5.1715**

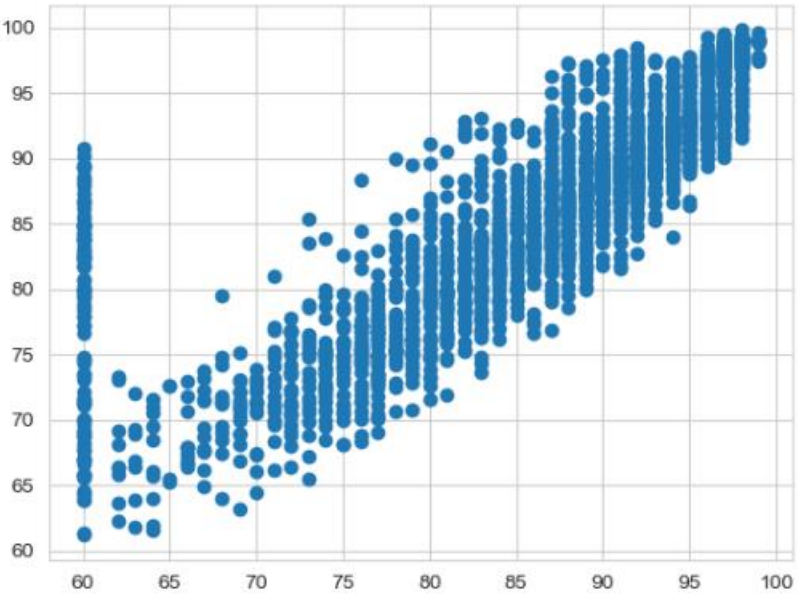
- **Scatterplot on Predicted Train and Test data:**

Figure 32: Scatter plot showing Predicted Train and Test data

Train data:



Test data:



- **Regression equation is determined as follows:**

(88.98) * const + (-0.06) * lread + (0.02) * lwrite + (-0.0) * scall + (-0.0) * sread + (-0.01) * swrite + (-0.1) * fork + (-0.32) * exec + (-0.0) * rchar + (-0.0) * wchar + (-0.29) * pgout + (0.03) * ppgout + (0.01) * pgfree + (-0.0) * pgscan + (0.06) * atch + (-0.08) * pgin + (0.01) * ppgin + (-0.02) * pflt + (-0.01) * vflt + (0.0) * freemem + (0.0) * freeswap + (2.13) * runqsz_Not_CPU_Bound +

- **Checking for Multicollinearity (VIF):**

Table 11: Table showing VIF of independent variables

```
lread ---> 6.546975816211261
lwrite ---> 4.481089649117559
scall ---> 9.35576279184697
sread ---> 19.270379440734164
swrite ---> 16.343355756996427
fork ---> 16.57081905169149
exec ---> 4.140547939116197
rchar ---> 4.275582278861992
wchar ---> 2.698968054952935
pgout ---> 6.622622899019428
ppgout ---> 11.9924436430458
pgfree ---> 11.391010029883274
pgscan ---> 4.735707224300264
atch ---> 1.895773594762528
pgin ---> 13.629760951475378
```

Here, 1 represents no multicollinearity, =>5 represents moderate multicollinearity and =>10 represents high multicollinearity. However, we can reduce the collinearity by removing the variables that does not impact r-squared values and are with high VIF.

Inference:

- OLS model seems to be a better option when compared with the other model as the difference between RMSE on train and test is slightly lower in OLS model.
- Even R-squared value is slightly higher in the OLS model.
- VIF in OLS model helps in identifying the variables that causes multicollinearity issues.
- OLS model also explains F-statistic, Skewness, Kurtosis and other measures.

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

Please explain and summarise the various steps performed in this project. There should be proper business interpretation and actionable insights present.

Ans:

- The average number reads between system memory and user memory is 19.56, whereas average number writes between system memory and user memory is 13.11.
- The average number of characters transferred per second by system read calls is 1.973857e+05, whereas, 9.590299e+04 is the average number of characters transferred per second by system write calls.

- Here, usr column has additional zeroes which can have a slight negative impact on the model. However, as per the model, the average Portion of time (%) that cpus run in user mode is 83.97.
- Based on the VIF, we can reduce the collinearity by removing the variables that does not impact r-squared values and have high VIF, this method helps in increase the model performance.
- We can remove the less importance variables such as ‘pgin’, ‘swrite’, ‘ppgout’, ‘pgfree’, ‘fork’, ‘sread’ and ‘scall’, in order to identify the attributes that affects the system to be in ‘usr’ mode.
- Attributes such as ‘atch’, ‘wchar’, ‘pgscan’ ‘exec’, ‘rchar’, ‘lwrite’ are the key players.
- The following linear equation helps the business in understanding the importance of each attribute that needs to be taken care of, and the proportion of mathematical equation to be considered.

```
(88.98) * const + (-0.06) * lread + (0.02) * lwrite + (-0.0) * scall + (-0.0) * sread + (-0.01) * swrite + (-0.1) * fork + (-0.
32) * exec + (-0.0) * rchar + (-0.0) * wchar + (-0.29) * pgout + (0.03) * ppgout + (0.01) * pgfree + (-0.0) * pgscan + (0.06) *
atch + (-0.08) * pgin + (0.01) * ppgin + (-0.02) * pflt + (-0.01) * vflt + (0.0) * freemem + (0.0) * freeswap + (2.13) * runqsz
_Not_CPU_Bound +
```

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.

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**.**

Ans:

- **Table 12: Top 5 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_exposure	Contraceptive_method_used
0	24.0	Primary	Secondary	3.0	Scientology	No	2	High	Exposed	No
1	45.0	Uneducated	Secondary	10.0	Scientology	No	3	Very High	Exposed	No
2	43.0	Primary	Secondary	7.0	Scientology	No	3	Very High	Exposed	No
3	42.0	Secondary	Primary	9.0	Scientology	No	3	High	Exposed	No
4	36.0	Secondary	Secondary	8.0	Scientology	No	3	Low	Exposed	No

- **Table 13: Bottom 5 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_exposure	Contraceptive_method_used
33.0	Tertiary	Tertiary	NaN	Scientology	Yes	2	Very High	Exposed	Yes
33.0	Tertiary	Tertiary	NaN	Scientology	No	1	Very High	Exposed	Yes
39.0	Secondary	Secondary	NaN	Scientology	Yes	1	Very High	Exposed	Yes
33.0	Secondary	Secondary	NaN	Scientology	Yes	2	Low	Exposed	Yes
17.0	Secondary	Secondary	1.0	Scientology	No	2	Very High	Exposed	Yes

- There are 1473 rows and 10 columns in the given data.

```
There are 1473 rows and 10 columns.
```

- **Info:**

Table 14: Basic info of the dataset

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1473 entries, 0 to 1472
Data columns (total 10 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   Wife_age                  1402 non-null   float64
 1   Wife_ education           1473 non-null   object
 2   Husband_education         1473 non-null   object
 3   No_of_children_born       1452 non-null   float64
 4   Wife_religion             1473 non-null   object
 5   Wife_Working              1473 non-null   object
 6   Husband_Occupation        1473 non-null   int64
 7   Standard_of_living_index  1473 non-null   object
 8   Media_exposure            1473 non-null   object
 9   Contraceptive_method_used 1473 non-null   object
dtypes: float64(2), int64(1), object(7)
memory usage: 115.2+ KB
```

Observations:

- There are 1473 rows and 10 columns.

- There are 2 float64, 1 int64 and 7 object datatypes.

- There are missing values for 2 columns (wife_age and no_of_children_born).

- **Duplicate values**: There are **80 duplicate rows** in the given dataset.

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

- We need to drop the duplicate values, as it affects the accuracy of the model.
- Dropping duplicate rows

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

- **Null values:**

Table 15: Table showing missing value info in the dataset

```
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
```

Observations:

- After removing duplicates, there are 67 missing values in ‘wife_age’ column and 21 missing values in ’no_of_children_born’ column.

- **Imputing null values:** We can impute the null values with mean or median for numeric variables and mode can be used for categorical variables. However, we are proceeding the imputation for missing values with Median, as number of children born is an integer data type.

Table 16: Table showing null values after median imputation

```
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
```

- **Statistical summary of numerical and categorical data:**

Table 17: Table showing statistical summary of numerical and categorical data

	count	mean	std	min	25%	50%	75%	max
Wife_age	1393.0	32.530510	8.088188	16.0	26.0	32.0	38.0	49.0
No_of_children_born	1393.0	3.286432	2.381791	0.0	1.0	3.0	5.0	16.0
Husband_Occupation	1393.0	2.174444	0.854590	1.0	1.0	2.0	3.0	4.0

	count	unique	top	freq
Wife_ education	1393	4	Tertiary	515
Husband_education	1393	4	Tertiary	827
Wife_religion	1393	2	Scientology	1186
Wife_Working	1393	2	No	1043
Standard_of_living_index	1393	4	Very High	618
Media_exposure	1393	2	Exposed	1284
Contraceptive_method_used	1393	2	Yes	779

Observations:

- Wife age ranges from minimum of 16 to maximum of 49.

- Number of children born ranges from minimum of 0 to maximum of 16.

- About 1043 of wives are not working.

- About 1284 of them are exposed to media.

- About 779 use contraceptive method.

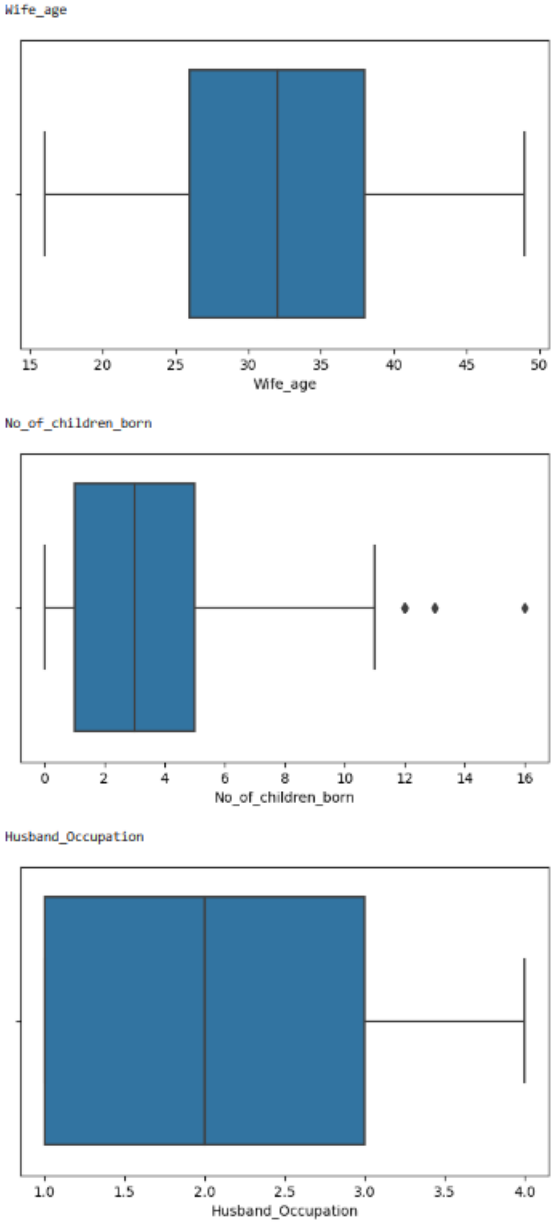
- **Unique values of categorical data:**

Table 18: Table showing unique values of categorical data

```
WIFE_ EDUCATION :  4
Uneducated     150
Primary        330
Secondary      398
Tertiary       515
Name: Wife_ education, dtype: int64


HUSBAND_EDUCATION :  4
Uneducated      44
Primary        175
Secondary      347
Tertiary       827
Name: Husband_education, dtype: int64


WIFE_RELIGION :  2
Non-Scientology     207
Scientology        1186
Name: Wife_religion, dtype: int64


WIFE_WORKING :  2
Yes      350
No      1043
Name: Wife_Working, dtype: int64


STANDARD_OF_LIVING_INDEX :  4
Very Low      129
Low           227
High          419
Very High     618
Name: Standard_of_living_index, dtype: int64


MEDIA_EXPOSURE  :  2
Not-Exposed     109
Exposed        1284
Name: Media_exposure , dtype: int64


CONTRACEPTIVE_METHOD_USED :  2
No      614
Yes     779
Name: Contraceptive_method_used, dtype: int64
```

- **Outliers:**

Figure 33: Boxplots after treating outliers



Observations:

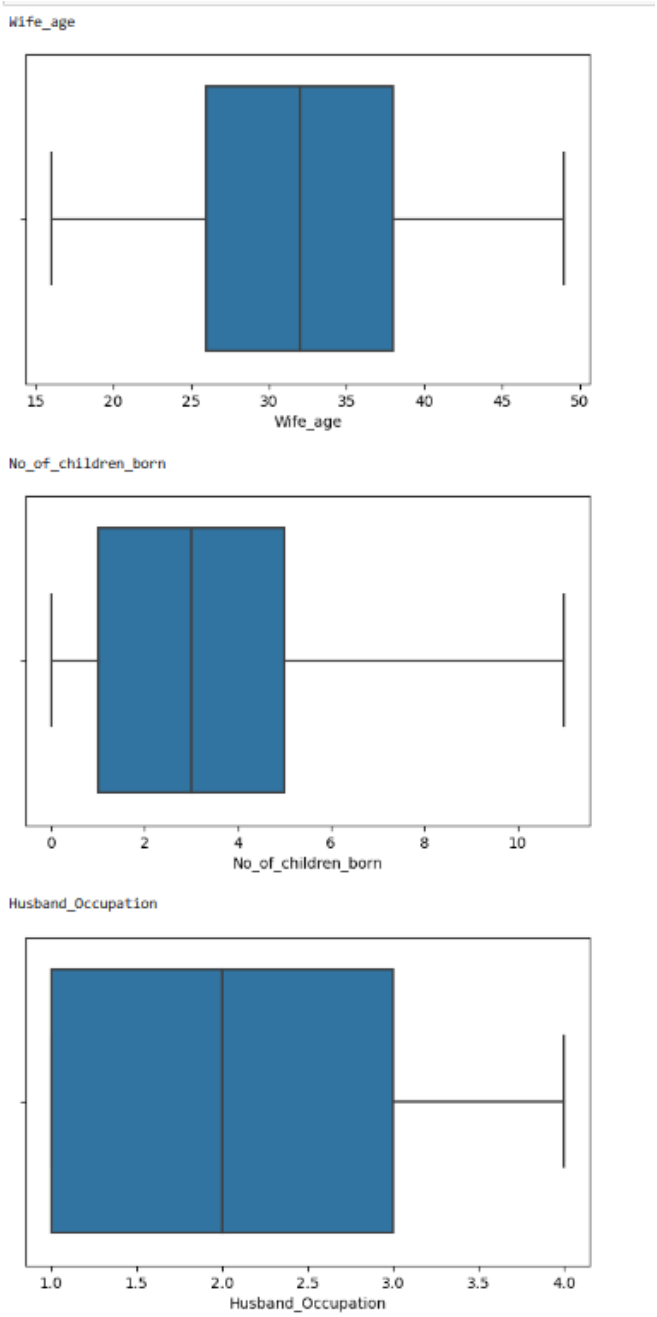
- There are outliers in ‘No of children born’ column.

- ‘Wife age’ column does not have any outliers.

- **Treating outliers**: We are treating outliers using the IQR method, where, extreme values are capped to the upper limit or whisker of the boxplot.

```
Lower Range : -5.0
Upper Range : 11.0
```

Figure 34: Boxplots after treating outliers

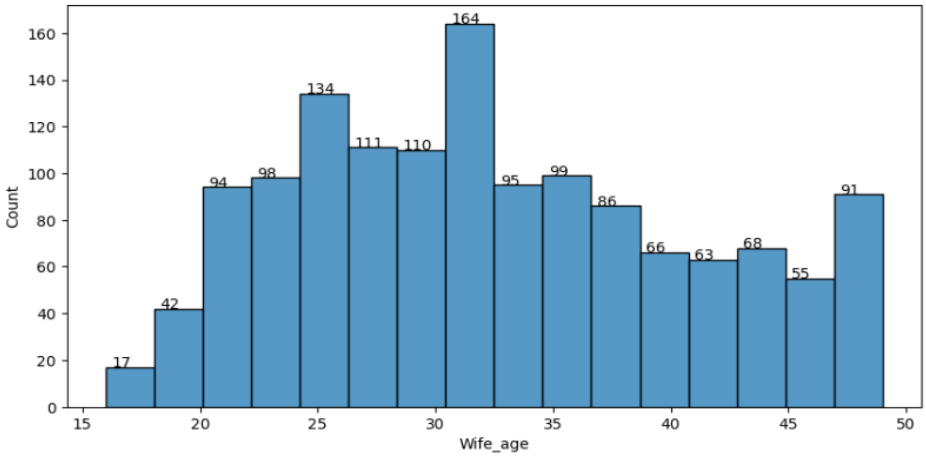


- Outliers have been treated.

- **<u>Univariate Analysis:</u>**

a) Wife age:

Figure 35: Histplot showing count of wife age



Statistical summary of the variable

```
count    1393.000000
mean       32.530510
std         8.088188
min        16.000000
25%        26.000000
50%        32.000000
75%        38.000000
max        49.000000
Name: Wife_age, dtype: float64
```

Observations

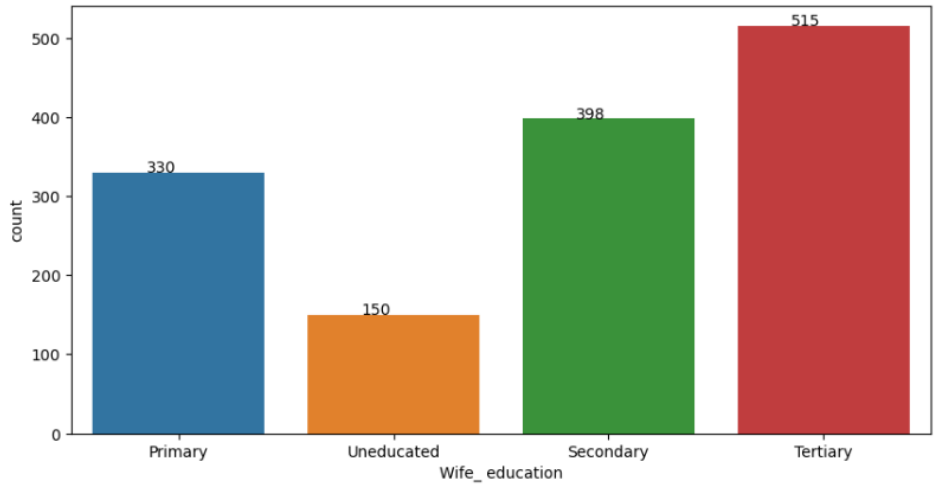
- The average number of wife age is 32.53.

- Wife age ranges from 16 to 49.

- About 75% of the wife age fall under 38.

b) Wife education:

Figure 36: Countplot showing wife education



Observations

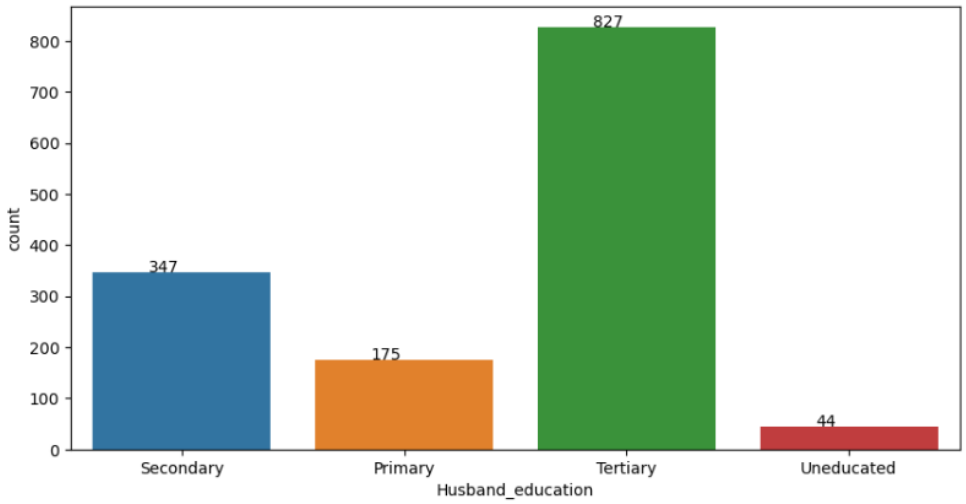
- Most of the wives have Tertiary education.

- 150 of 1393 wives are uneducated.

- 330 wives have Primary education and 398 have Secondary education.

c) Husband education:

Figure 37: Countplot showing husband education



Observations

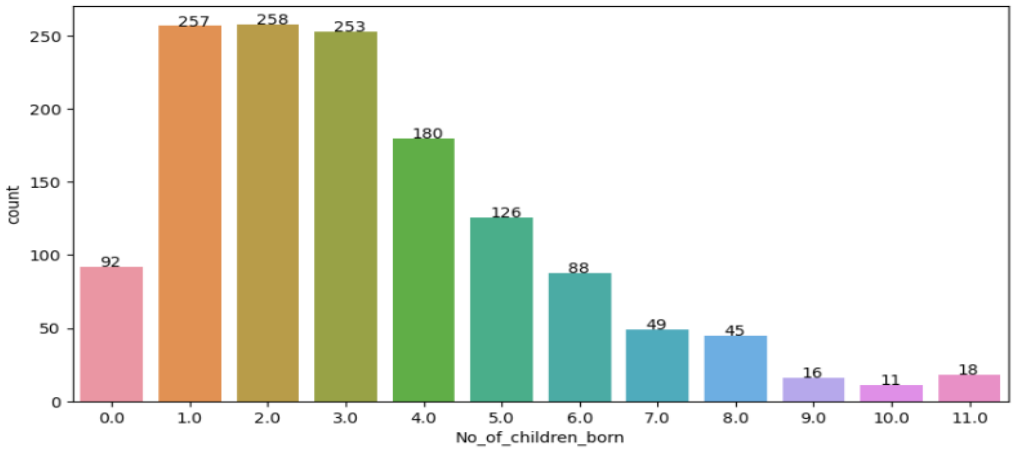
- About 44 of 1393 husbands are uneducated.

- 827 of 1393 husbands have Tertiary education.

- 175 husbands have Primary education and 347 have Secondary education.

d) Number of children born:

Figure 38: Countplot showing number of children born



Statistical summary of the variable

```
count    1393.000000
mean        3.277100
std         2.345673
min         0.000000
25%         1.000000
50%         3.000000
75%         5.000000
max        11.000000
Name: No_of_children_born, dtype: float64
```

Observations

- Average number of children born are 3.

- Number of children born ranges from 0 to 11.

- About 258 of 1393 wives have 2 children.

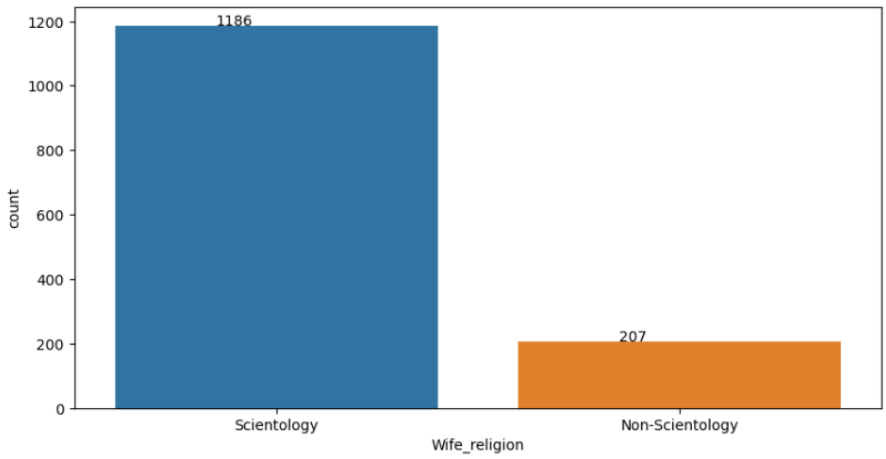
- About 257 of 1393 wives have 1 child.

- About 92 of 1393 wives have no children.

- About 11 wives have 10 children.

e) Wife religion:

Figure 39: Countplot showing wife religion



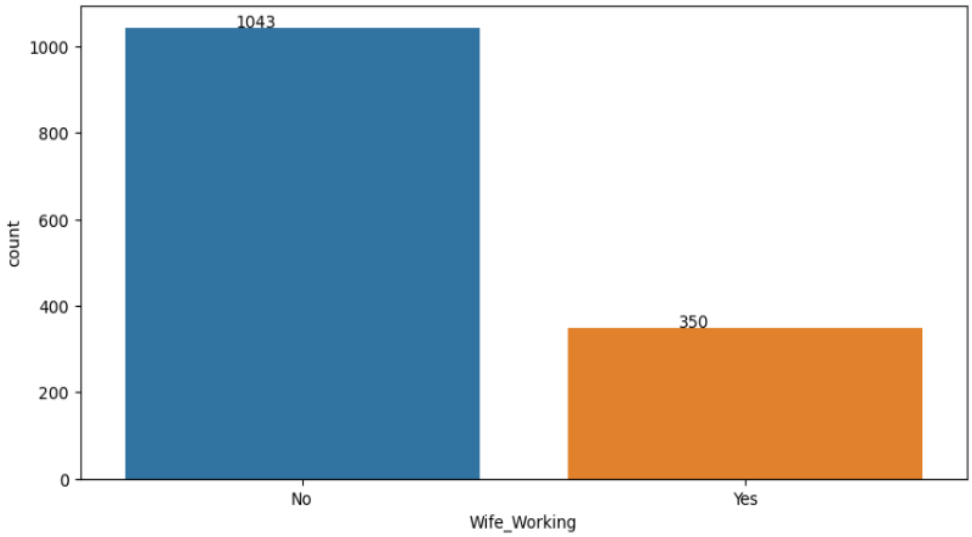
Observations

- About 1186 of 1393 wives are Scientology.

- About 207 of 1393 wives are non-Scientology.

f) Wife working:

Figure 40: Countplot showing wife working



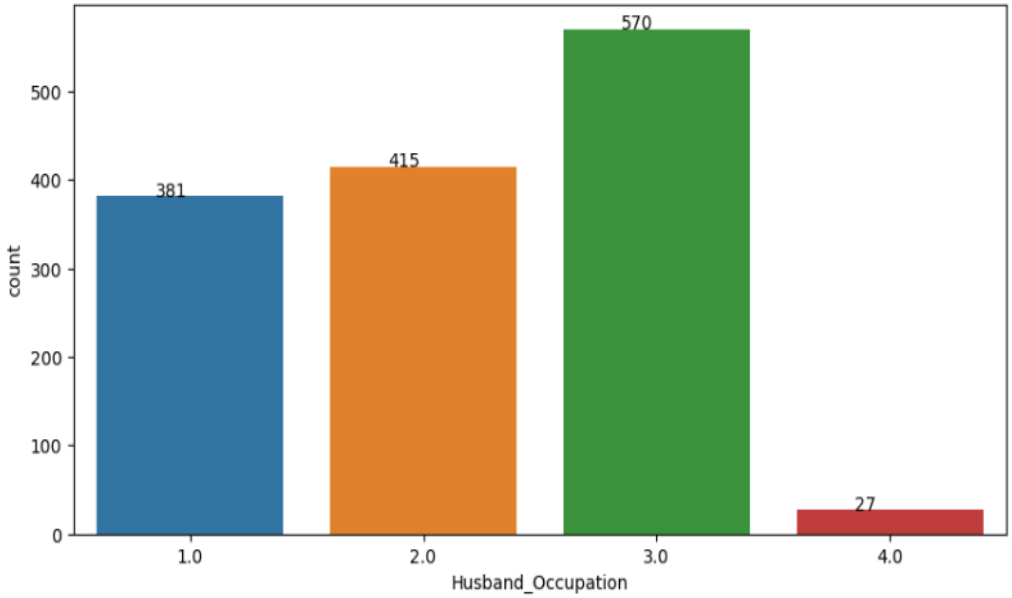
Observations

- About 1043 of 1393 wives are working.

- About 350 wives are not working.

g) Husband occupation:

Figure 41: Countplot showing husband occupation

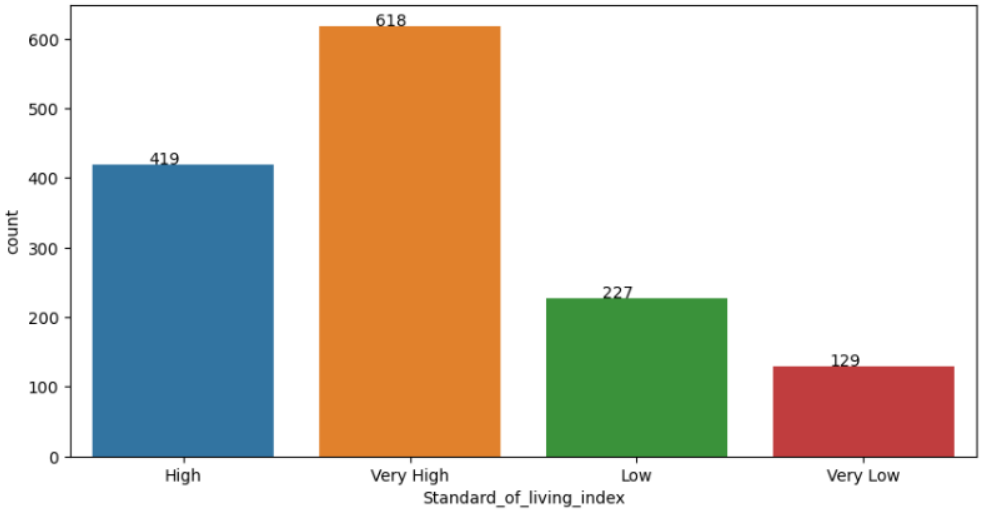


Observations

- About 27 of 1393 husbands have very high occupation.

h) Standard of living index:

Figure 42: Countplot showing standard of living index



Observations

- 129 of 1393 have very low standard of living index.

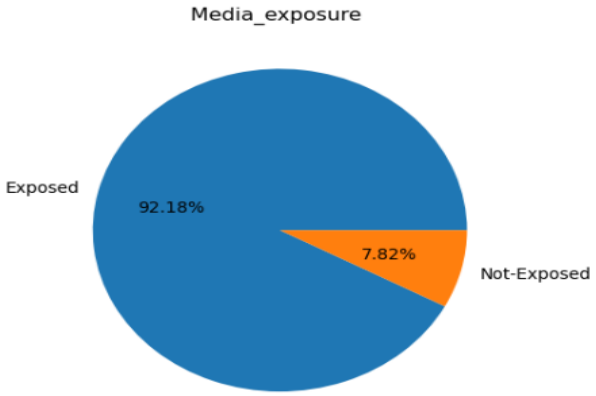
- 618 of 1393 have very high standard of living index.

- 419 of 1393 have high standard of living index.

- 227 of 1393 have low standard of living index.

i) Media exposure:

Figure 43: Pie-chart showing media exposure



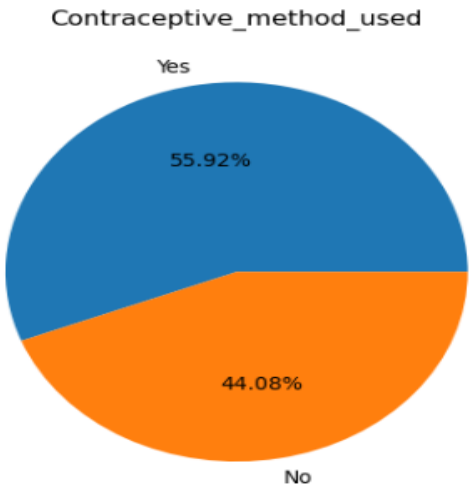
Observations

- 92.18% of wives are exposed to Media.

- 7.82% of wives are not exposed to Media.

j) Contraceptive method used:

Figure 44: Pie-chart showing contraceptive method used



Observations

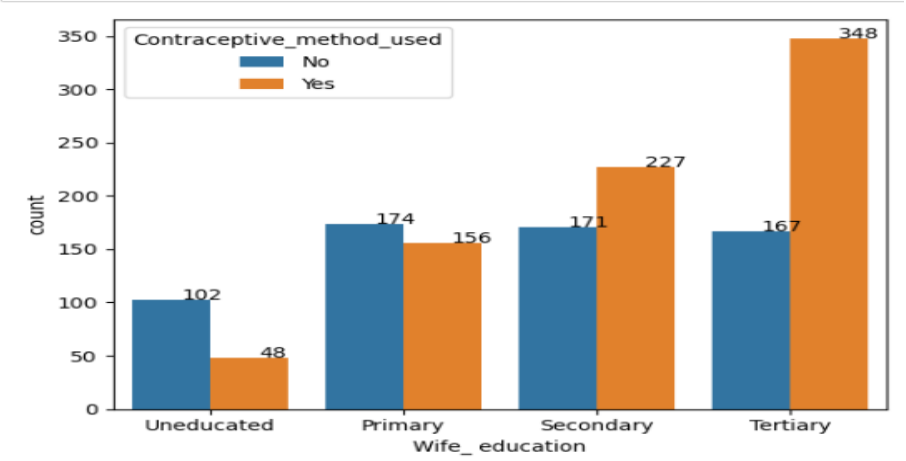
- 55.92% of wives use contraceptive method.

- 44.09% of wives do not use contraceptive method.

- **<u>Bivariate and Multivariate Analysis:</u>**

a) Wife education and Contraceptive method used:

Figure 45: Countplot showing Wife education and Contraceptive method used



Observations

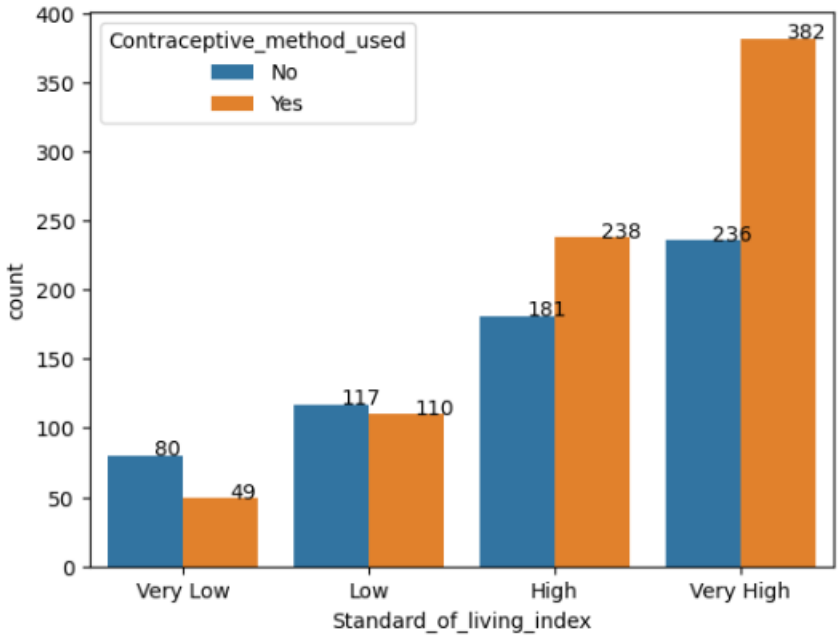
- The above countplot states that wives with Tertiary education use contraceptive method much higher than the other groups.

- Uneducated wives are the less users of contraceptive method.

- Education plays a vital role in the usage of contraceptive methods.

b) Standard_of_living_index and Contraceptive_method_used:

Figure 46: Countplot showing standard of living index and contraceptive method used



Observations

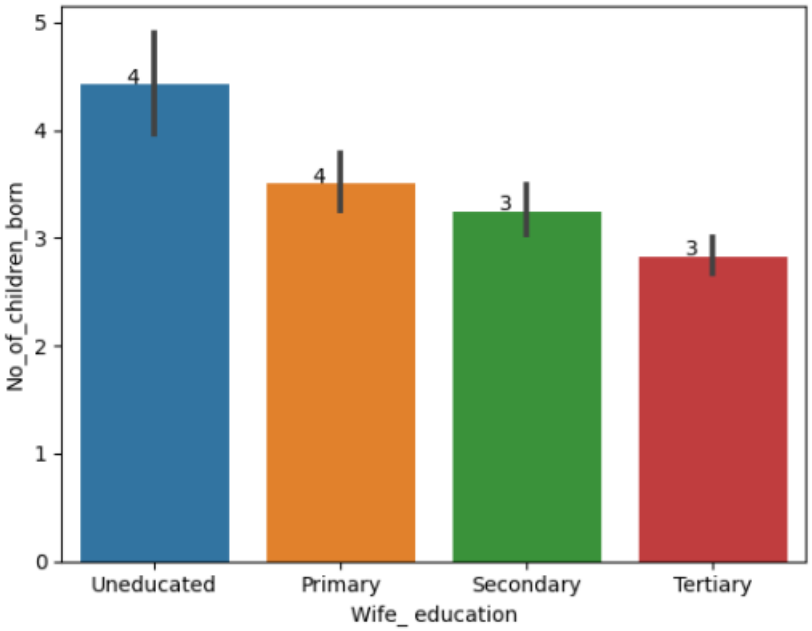
- Wives with very low standard of living use contraceptive method in a very less manner.

- The above countplot states that wives with very high standard of living use contraceptive method much higher than the other groups.

- Standard of living also plays a vital role in the usage of contraceptive methods.

c) Wife education and No_of_children_born:

Figure 47: Barplot showing Wife education and number of children born



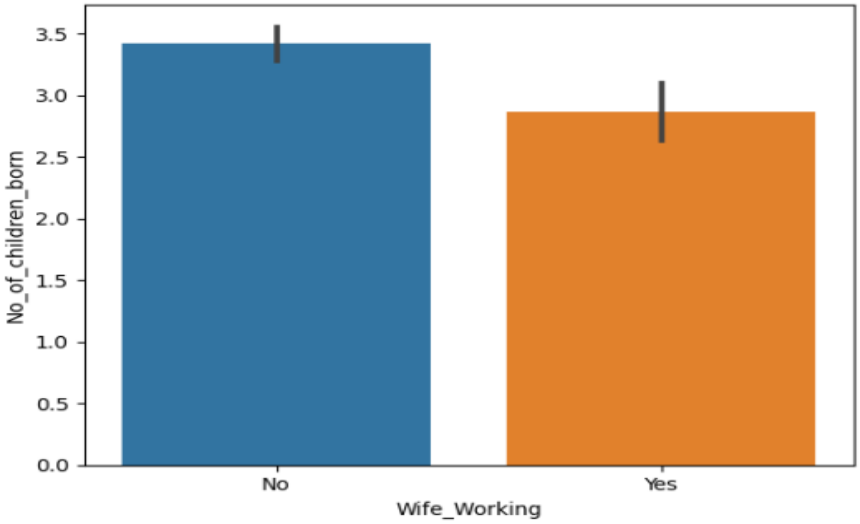
Observations

- The above barplot states that wives who are uneducated have a greater number of children.

- Wives with tertiary education have a smaller number of children when compared with other groups.

d) Wife working and No_of_children_born:

Figure 48: Barplot showing Wife working and number of children born



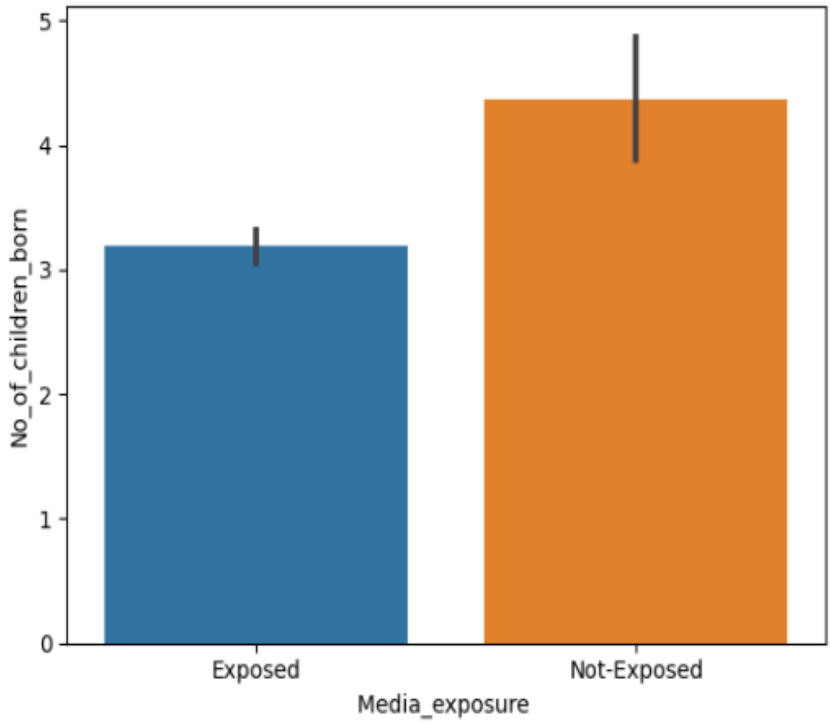
Observations

- The above barplot states that the wives who are working have a smaller number of children.

- Wives who are not working have a greater number of children.

e) Media exposure and No_of_children_born:

Figure 49: Barplot showing media exposure and number of children born



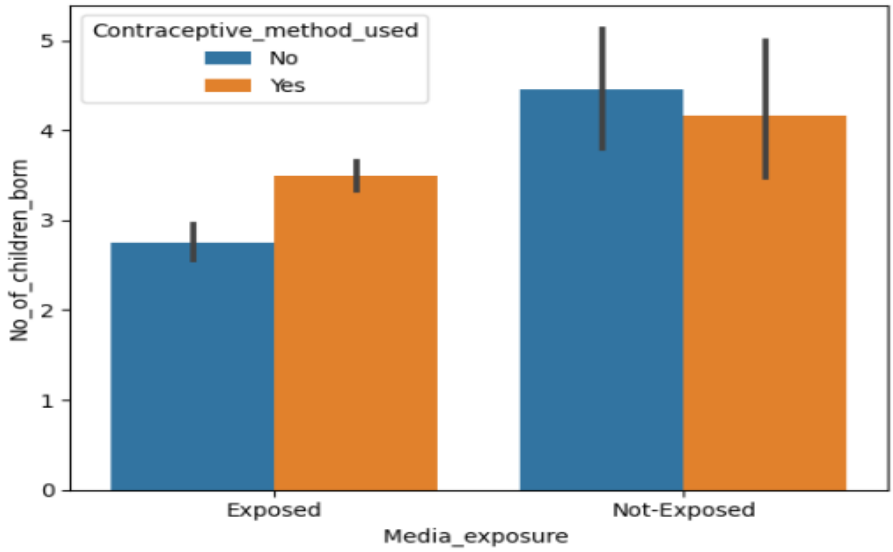
Observations

- The above barplot states that the wives who are exposed to media have a smaller number of children.

- Wives who do not have media exposure have a greater number of children.

f) Media exposure, No_of_children_born& Contraceptive_method_used:

Figure 50: Barplot showing media exposure, number of children born and contraceptive method used



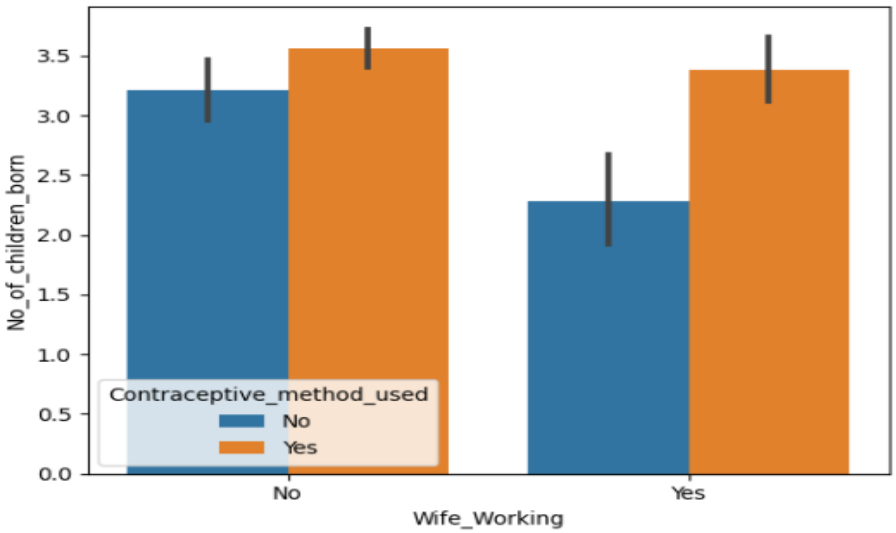
Observations

- The above barplot states that the wives who are exposed to media have a smaller number of children and are the users of contraceptive method in a high manner than that of non-users.

- Wives who do not have media exposure have a greater number of children and are the less users of contraceptive method comparatively.

g) Wife working, No_of_children_born and Contraceptive_method_used:

Figure 51: Barplot showing Wife working, number of children born and contraceptive method used



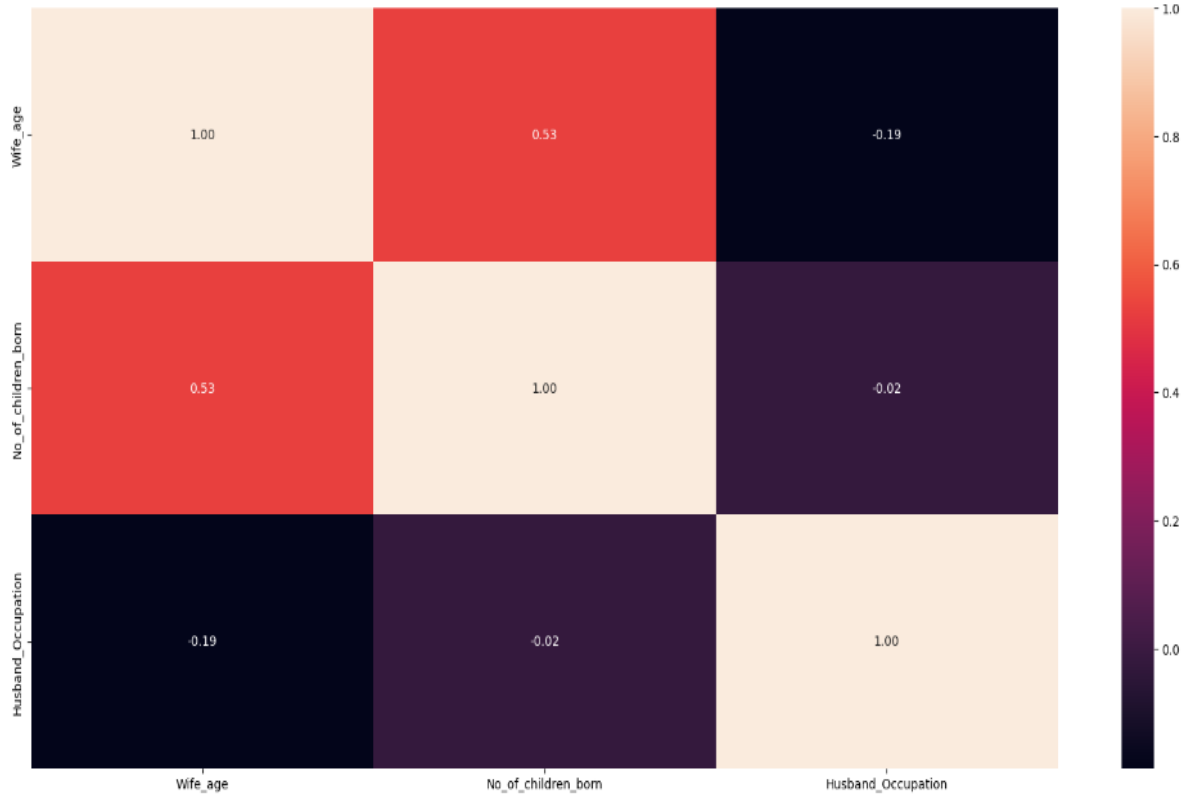
Observations

- The above barplot states that the wives who are working have a smaller number of children and are the more users of contraceptive methods comparatively.

- Wives who are not working have a greater number of children and are the more users of contraceptive methods than the non-users.

h) Correlation between the variables:

Figure 52: Heatmap showing correlation between variables



Observations

- Wife age and number of children are positively correlated i.e., by 0.53.

- Wife age and husband occupation are negatively correlated i.e., by -0.19.

- Number of children and husband occupation are weakly related i.e., by -0.02 which means there is a negative relationship between the variables.

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.

Ans:

- Initially, encoding the target class (Contraceptive_method_used) with 0 for 'No's and 1 for 'Yeses'.
- Converting other 'object' type variables to dummy variables.

Table 19: Table showing dummy variables

	Wife_age	No_of_children_born	Husband_Occupation	Contraceptive_method_used	Wife_education_Secondary	Wife_education_Tertiary	Wife_education_Uneducated
0	24.0	3.0	2.0	0	0	0	0
1	45.0	10.0	3.0	0	0	0	1
2	43.0	7.0	3.0	0	0	0	0
3	42.0	9.0	3.0	0	1	0	0
4	36.0	8.0	3.0	0	1	0	0

Wife_religion_Scientology	Wife_Working_Yes	Standard_of_living_index_Low	Standard_of_living_index_Very High	Standard_of_living_index_Very Low	Media_exposure_Not-Exposed
1	0	0	0	0	0
1	0	0	1	0	0
1	0	0	1	0	0
1	0	0	0	0	0
1	0	1	0	0	0

- Assigning all independent variables to 'X'.
- Assigning dependent variables to 'Y'.
- Splitting the data into train and test at 70:30 ratio.

a) **Logistic Regression Model:**

- Initializing and Fitting the **Logistic Regression Model**

```
[Parallel(n_jobs=2)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=2)]: Done   1 out of   1 | elapsed:    1.1s finished
```

```
LogisticRegression(max_iter=10000, n_jobs=2, penalty='none', solver='newton-cg',
                   verbose=True)
```

- Predicting on Training and Test dataset

Table 20: Table showing probability prediction on X train dataset

```
array([[0.17441285, 0.82558715],
       [0.46271278, 0.53728722],
       [0.69413704, 0.30586296],
       ...,
       [0.33503026, 0.66496974],
       [0.7145228 , 0.2854772 ],
       [0.60967449, 0.39032551]])
```

- Predicted classes and Probability

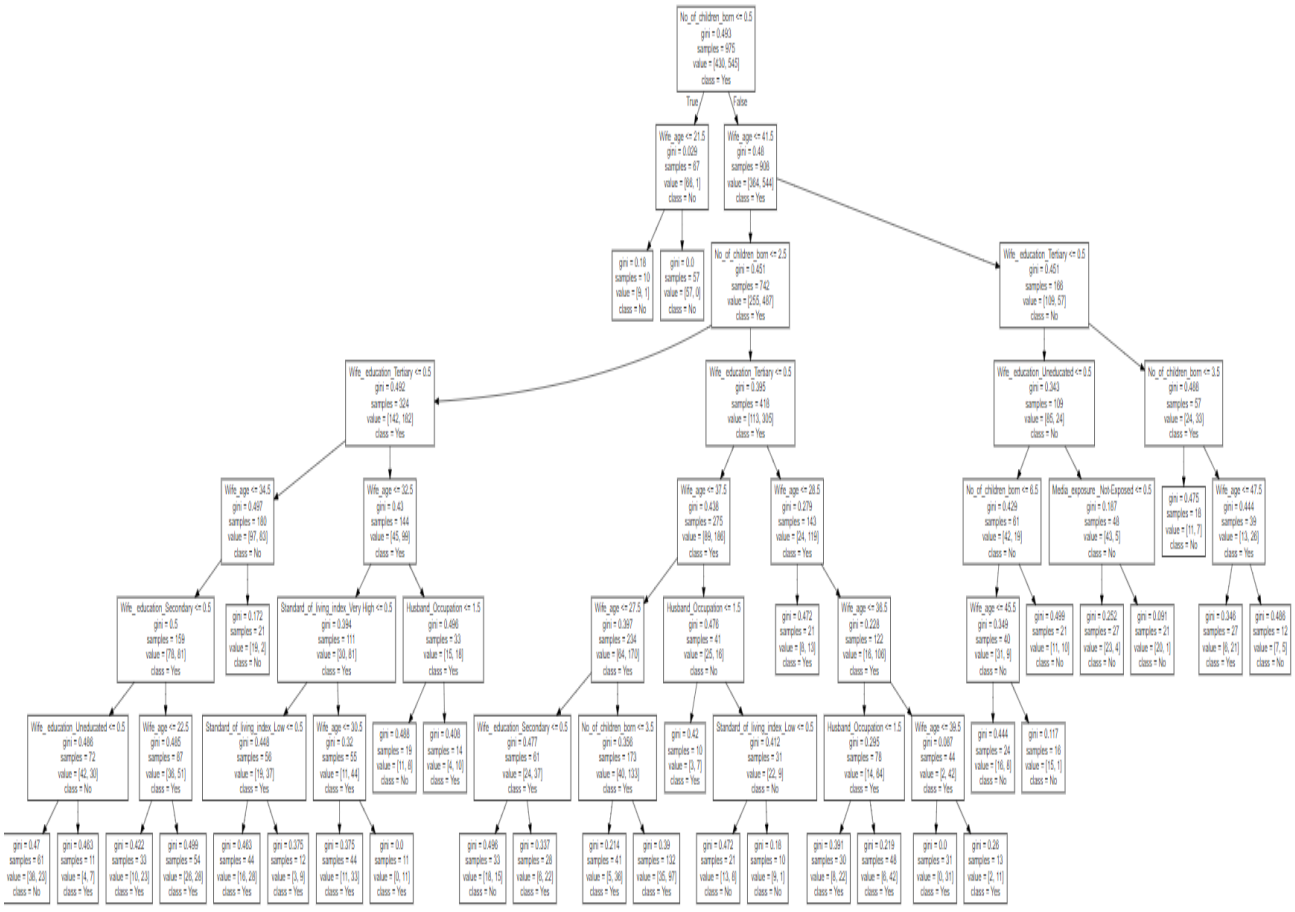
Table 21: Table showing predicted classes and Probability

	0	1
0	0.183460	0.816540
1	0.712886	0.287114
2	0.287220	0.712780
3	0.583461	0.416539
4	0.450613	0.549387

b) **CART:**

- Initializing and Fitting **CART Model.**
- Building a decision tree classifier by creating a dot file of Max depth 7.

Table 22: Table showing regularized decision tree of depth 7



- **Importance of variables and their probabilities of importance**

Table 23: Table showing probability of Importance of variables

	Imp
Wife_age	0.341565
No_of_children_born	0.222627
Husband_Occupation	0.083771
Wife_ education_Secondary	0.028566
Wife_ education_Tertiary	0.034139
Wife_ education_Uneducated	0.028235
Husband_education_Secondary	0.036812
Husband_education_Tertiary	0.015925
Husband_education_Uneducated	0.006004
Wife_religion_Scientology	0.031070
Wife_Working_Yes	0.040677
Standard_of_living_index_Low	0.037347
Standard_of_living_index_Very High	0.044472
Standard_of_living_index_Very Low	0.028928
Media_exposure _Not-Exposed	0.019862

- We can keep or remove variables from the model based on their importance.

- **Predicted classes**

Table 24: Table showing predicted classes of Train data

```
array([1, 0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1,
       1, 1, 0, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 0, 1, 0, 0, 1, 1, 0,
       1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1,
       1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 0, 1, 1, 1, 0, 1, 1, 0, 1, 1, 0, 0,
       1, 1, 0, 1, 1, 0, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1,
       0, 1, 1, 0, 1, 1, 1, 1, 0, 1, 0, 1, 1, 0, 1, 1, 0, 1, 1, 1, 0, 1,
       0, 1, 0, 1, 1, 0, 1, 1, 0, 1, 1, 0, 1, 0, 1, 0, 1, 1, 1, 1, 0, 0,
       1, 0, 0, 1, 1, 1, 0, 1, 0, 0, 1, 0, 1, 1, 1, 1, 0, 0, 1, 1, 0, 0,
       1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1,
       1, 0, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 0, 1, 1, 0, 1, 1, 0, 0, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 0, 0,
       0, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 0, 1, 1, 0, 0, 1, 0, 0, 1, 1,
       1, 0, 0, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 0, 1, 0, 1, 1, 1,
       0, 1, 1, 0, 1, 1, 0, 0, 1, 0, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1,
       0, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 0, 1, 1, 1, 1, 1, 0, 0, 1,
       1, 1, 0, 1, 0, 1, 0, 1, 1, 1, 0, 0, 1, 1, 1, 1, 0, 1, 1, 0, 0, 0,
       0, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 1, 1, 1, 1,
       1, 0, 0, 1, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 1, 1,
       1, 1, 0, 1, 0, 0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 1, 0, 1, 0, 1, 0, 0,
       1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 0, 1, 1, 1, 0, 1, 0, 1,
       1, 0, 1, 1, 1, 1, 1, 0, 0, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 0, 1,
       0, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 0, 1, 0, 0, 1, 1, 0,
       0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1, 1,
       1, 0, 1, 0, 1, 1, 1, 1, 0, 1, 1, 0, 0, 1, 1, 0, 1, 0, 1, 0, 0, 1,
       1, 0, 0, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1,
       1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1,
       1, 1, 0, 0, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 0, 0, 1, 0,
       1, 0, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 0, 0, 1, 1, 0, 1, 1, 0, 1, 1,
       1, 0, 1, 0, 0, 1, 1, 0, 0, 1, 0, 0, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1,
       1, 0, 1, 0, 0, 0, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1,
       1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 0, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1,
       1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1,
       1, 1, 1, 0, 1, 0, 1, 1, 0, 0, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1,
       1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0,
       1, 1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 0, 0, 1,
       1, 1, 1, 0, 1, 0, 1, 0, 1, 1, 1, 0, 0, 0, 1, 1, 1, 0, 1, 1, 0, 1,
       1, 1, 0, 1, 0, 0, 0, 1, 0, 1, 0, 1, 1, 1, 1, 1, 0, 1, 0, 1, 0, 1,
       1, 1, 1, 0, 0, 1, 0, 1, 0, 0, 1, 1, 0, 0, 1, 0, 1, 1, 1, 0, 1, 0,
       1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 0, 0,
       0, 1, 1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1,
       1, 0, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1,
       0, 1, 1, 0, 1, 0, 0])
```

Table 25: Table showing predicted classes of Test data

```
array([1, 0, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 0, 1, 1, 0, 0, 1, 1, 1,
       0, 0, 1, 0, 1, 1, 0, 0, 1, 1, 1, 1, 1, 0, 1, 0, 0, 1, 1, 1, 0, 1,
       1, 1, 0, 0, 1, 1, 0, 1, 1, 1, 1, 0, 1, 0, 0, 1, 1, 1, 0, 0, 1, 1,
       1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 0, 1, 0, 1, 0, 1, 1, 1, 1, 0, 1, 1,
       0, 0, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0,
       1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1,
       1, 0, 0, 1, 1, 0, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0,
       1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1,
       1, 0, 1, 0, 1, 1, 0, 0, 0, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1,
       1, 1, 1, 1, 0, 1, 0, 1, 1, 0, 1, 1, 1, 0, 1, 1, 0, 1, 0, 1, 1, 0,
       1, 1, 0, 1, 0, 0, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 0, 0, 1, 1,
       1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1,
       1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 0, 0, 1, 1, 1, 0,
       1, 1, 1, 0, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 0, 1, 0, 0, 1, 0, 1, 0,
       0, 1, 0, 1, 0, 1, 0, 0, 1, 1, 0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 1,
       1, 1, 0, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 0, 1, 0, 0,
       0, 1, 1, 1, 0, 1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 0, 1,
       1, 1, 0, 0, 1, 0, 1, 0, 0, 0, 1, 1, 0, 1, 0, 1, 1, 0, 1, 0, 1, 1,
       1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 1, 1, 0, 1])
```

c) **LDA:**

- Initializing and Fitting **LDA Model.**
- Number of rows and columns of the training and testing dataset at 70:30 ratio are as follows:

Table 26: Table showing training and testing dataset at 70:30 ratio

```
Number of rows and columns of the training set for the independent variables: (975, 15)
Number of rows and columns of the training set for the dependent variable: (975,)
Number of rows and columns of the test set for the independent variables: (418, 15)
Number of rows and columns of the test set for the dependent variable: (418,)
```

- Predicting Train and Test dataset.
- Probability prediction for the training and test data

Table 27: Table showing probability prediction for the Training data

```
array([0.8322338 , 0.53614881, 0.31225187, 0.58313459, 0.54804258,
       0.34371614, 0.42531346, 0.71428564, 0.10355985, 0.71097206,
       0.20597382, 0.56949355, 0.35770806, 0.78646366, 0.42560393,
       0.68567738, 0.72965536, 0.35798068, 0.83721691, 0.63394752,
       0.81264764, 0.85159833, 0.55042976, 0.28709568, 0.42444162,
       0.6987824 , 0.23221266, 0.33593909, 0.79064197, 0.59850797,
       0.30637606, 0.81356943, 0.52852318, 0.80283108, 0.6985405 ,
       0.20146051, 0.81127408, 0.55697337, 0.40827556, 0.54400801,
       0.33575938, 0.68327019, 0.82055871, 0.37191551, 0.59964807,
       0.5651852 , 0.83523145, 0.15141921, 0.4922527 , 0.52392817,
       0.80754558, 0.7587188 , 0.82763401, 0.36730696, 0.66107265,
       0.76921557, 0.70561413, 0.78197881, 0.71306568, 0.24776669,
       0.71966187, 0.33124407, 0.70349303, 0.58656983, 0.72198561,
       0.78896635, 0.71839599, 0.4267619 , 0.68053315, 0.47754474,
       0.47008999, 0.82687743, 0.06047247, 0.11738214, 0.86301675,
       0.81004059, 0.83050941, 0.57333908, 0.85735895, 0.51521962,
       0.50227328, 0.53192552, 0.51602918, 0.21292159, 0.36422064,
       0.28658822, 0.47197818, 0.07078022, 0.57576351, 0.59950193,
       0.38566113, 0.72471122, 0.36137484, 0.46298211, 0.64612944,
```

Table 28: Table showing probability prediction for the Test data

```
array([0.81865279, 0.29602709, 0.71759732, 0.42702567, 0.5364965 ,
       0.70393918, 0.60584429, 0.57834007, 0.85015199, 0.38240825,
       0.67428395, 0.22966325, 0.42188339, 0.63937155, 0.17123075,
       0.81857431, 0.83023878, 0.47442487, 0.61343143, 0.89450691,
       0.31799393, 0.75695264, 0.4793547 , 0.20409907, 0.27767773,
       0.62161035, 0.82919502, 0.49373325, 0.69691008, 0.3074374 ,
       0.80414977, 0.85821305, 0.63867177, 0.57335395, 0.84255051,
       0.35224689, 0.48363488, 0.52648784, 0.25929516, 0.54999484,
       0.82539038, 0.6167401 , 0.63300499, 0.65860084, 0.83838436,
       0.48698851, 0.28223555, 0.03540808, 0.7459538 , 0.56702643,
       0.2913924 , 0.53732466, 0.59718618, 0.76886884, 0.76605874,
       0.43763581, 0.65816211, 0.23133881, 0.4585153 , 0.69207232,
       0.87520106, 0.51856369, 0.2861127 , 0.1823641 , 0.78346627,
       0.62957774, 0.64676641, 0.47432193, 0.09134582, 0.66458176,
       0.71023477, 0.60902056, 0.56651912, 0.02982132, 0.55737889,
       0.5900116 , 0.38346602, 0.60873392, 0.26105218, 0.24700092,
       0.52493018, 0.93797803, 0.45933612, 0.67475077, 0.90651976,
       0.59704794, 0.65754934, 0.35882109, 0.2528706 , 0.29230806,
       0.63037883, 0.37863116, 0.65666177, 0.72902174, 0.45217262,
```

- The above table shows the predicted classes of both training and 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.

Ans:

a) **Logistic Regression Model:**

- **Accuracy:**
  - Accuracy score of **Train** dataset is 0.6738
  - Accuracy score of **Test** dataset is 0.6340
- **Confusion Matrix and Classification report:**
  - **Train** data:

Table 29: Table showing confusion matrix for the Train data

```
array([[227, 203],
       [115, 430]],
```

Table 30: Table showing classification report matrix for the Train data

	precision	recall	f1-score	support
0	0.66	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

- **Test** data:

Table 31: Table showing confusion matrix for the Test data

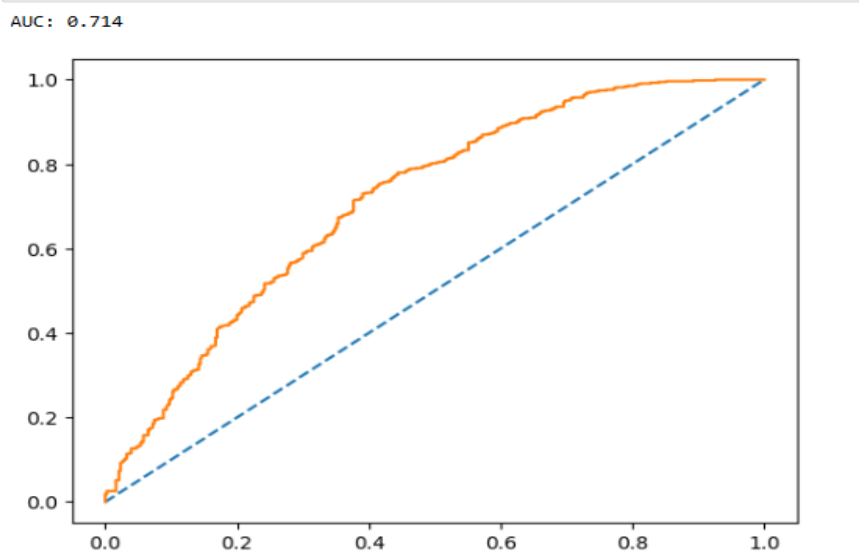
```
array([[ 91,  93],
       [ 60, 174]],
```

Table 32: Table showing classification report matrix for the Test data

	precision	recall	f1-score	support
0	0.60	0.49	0.54	184
1	0.65	0.74	0.69	234
accuracy			0.63	418
macro avg	0.63	0.62	0.62	418
weighted avg	0.63	0.63	0.63	418

- **ROC Curve & ROC_AUC score:**
  - **Train** data:

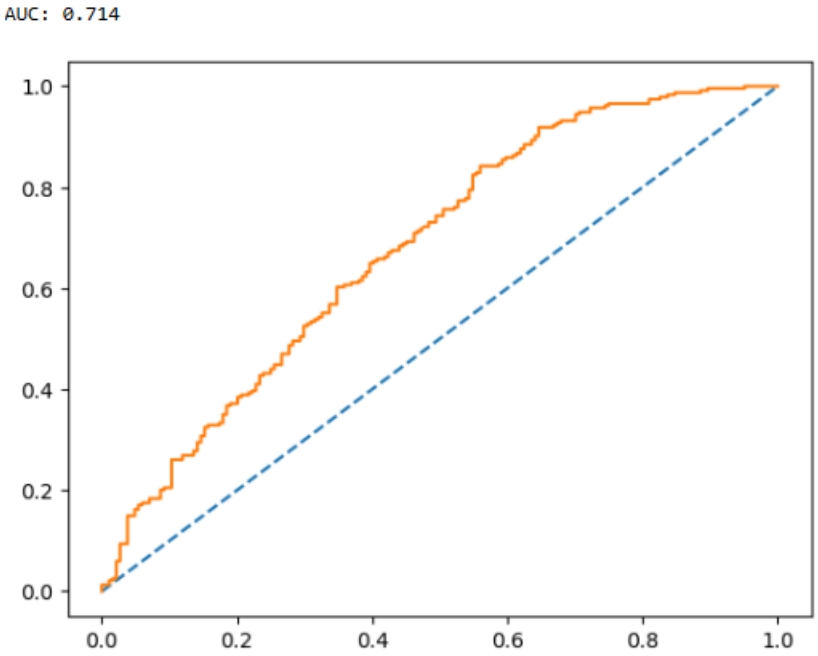
Figure 53: Figure showing ROC curve and ROC_AUC score for the Train data



- **ROC_AUC score for Train data is 0.714.**

- **Test** data:

Figure 54: Figure showing ROC curve and ROC_AUC score for the Test data



- **ROC_AUC score for Test data is 0.714.**

b) **CART:**

- **Accuracy:**
  - Accuracy score of **Train** dataset is 0.7415
  - Accuracy score of **Test** dataset is 0.6699
- **Confusion Matrix and Classification report:**
  - **Train** data:

Table 33: Table showing confusion matrix for the Train data

```
array([[248, 182],
       [ 70, 475]],
```

Table 34: Table showing classification report matrix for the Train data

	precision	recall	f1-score	support
0	0.78	0.58	0.66	430
1	0.72	0.87	0.79	545
accuracy			0.74	975
macro avg	0.75	0.72	0.73	975
weighted avg	0.75	0.74	0.73	975

- **Test** data:

Table 35: Table showing confusion matrix for the Test data

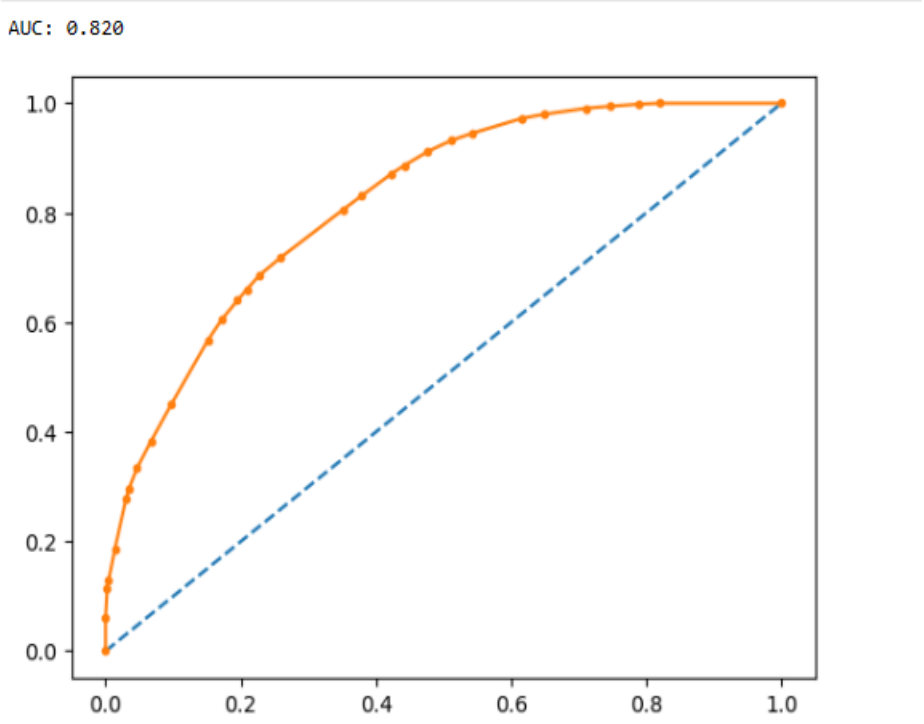
```
array([[ 97,  87],
       [ 51, 183]],
```

Table 36: Table showing classification report matrix for the Test data

	precision	recall	f1-score	support
0	0.66	0.53	0.58	184
1	0.68	0.78	0.73	234
accuracy			0.67	418
macro avg	0.67	0.65	0.66	418
weighted avg	0.67	0.67	0.66	418

- **ROC Curve & ROC_AUC score:**
  - **Train** data:

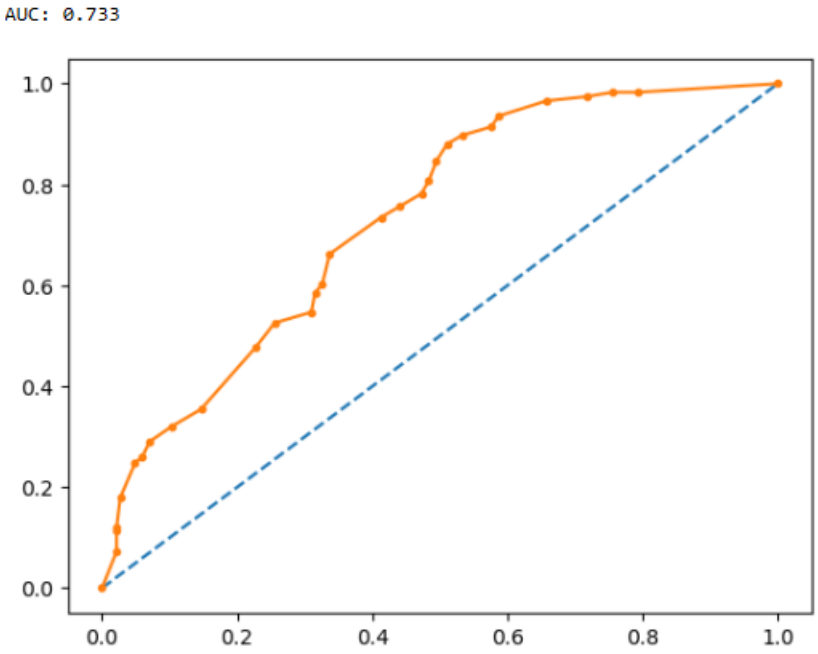
Figure 55: Figure showing ROC curve and ROC_AUC score for the Train data



- **ROC_AUC score for Train data is 0.820**

- **Test** data:

Figure 56: Figure showing ROC curve and ROC_AUC score for the Test data



- **ROC_AUC score for Test data is 0.733**
-

c) **LDA:**

- **Accuracy:**
  - Accuracy score of **Train** dataset is 0.6769
  - Accuracy score of **Test** dataset is 0.6340
- **Confusion Matrix and Classification report:**
  - **Train** data:

Table 37: Table showing confusion matrix for the Train data

```
array([[224, 206],
       [109, 436]],
```

Table 38: Table showing classification report matrix for the Train data

Classification Report of the training data:

	precision	recall	f1-score	support
0	0.67	0.52	0.59	430
1	0.68	0.80	0.73	545
accuracy			0.68	975
macro avg	0.68	0.66	0.66	975
weighted avg	0.68	0.68	0.67	975

- **Test** data:

Table 39: Table showing confusion matrix for the Test data

```
array([[ 87,  97],
       [ 56, 178]],
```

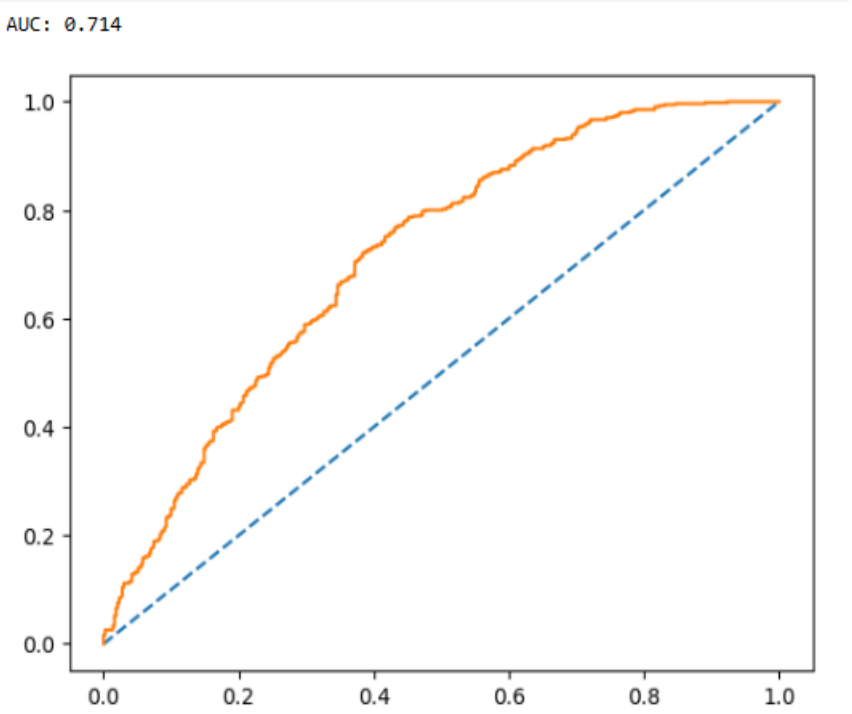
Table 40: Table showing classification report matrix for the Test data

Classification Report of the test data:

	precision	recall	f1-score	support
0	0.61	0.47	0.53	184
1	0.65	0.76	0.70	234
accuracy			0.63	418
macro avg	0.63	0.62	0.62	418
weighted avg	0.63	0.63	0.63	418

- **ROC Curve & ROC_AUC score:**
  - **Train** data:

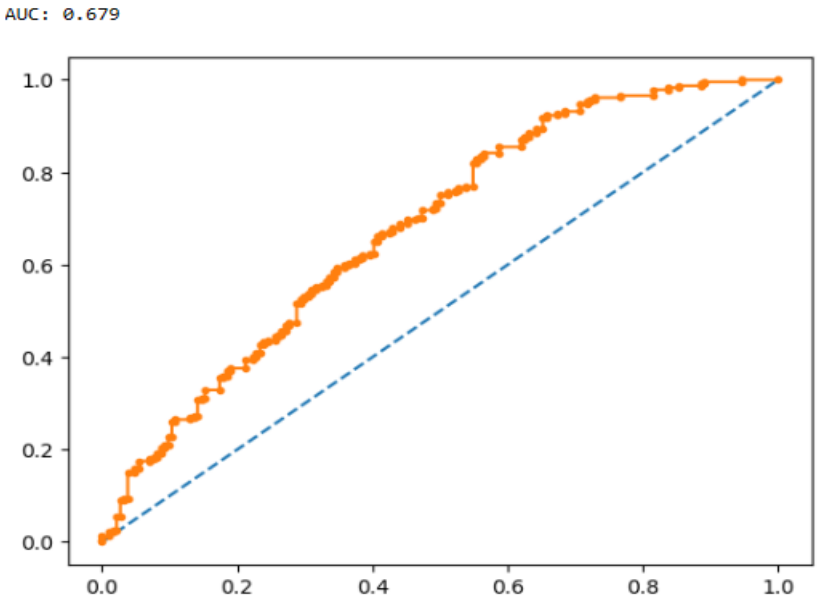
Figure 57: Figure showing ROC curve and ROC_AUC score for the Train data



- **ROC_AUC score for Train data is 0.714.**

- **Test** data:

Figure 58: Figure showing ROC curve and ROC_AUC score for the Test data



- **ROC_AUC score for Test data is 0.679.**

Inference:

- CART or decision tree would be the best model when compared with the other models.
- Even though, we cannot rely on accuracy score, CART has the highest accuracy score with 74.15% on Train data and 67% on Test data, which is better when compared with the other models.
    - **For contraceptive method not used,**
- CART has precision of 78% on the Train data, which is higher than LDA 67% and Logistic Regression 66%.
- CART has precision of 66% on the Test data, which is higher than LDA 61% and Logistic Regression 60%.
- CART has recall of 58% on the Train data, and 53% on Test data which is higher than other models.
    - **For contraceptive method used,**
- CART has precision of 72% on the Train data, which is higher than LDA 68% and Logistic Regression 68%.
- CART has precision of 68% on the Test data, which is higher than LDA 65% and Logistic Regression 65%.
- CART has recall of 87% on the Train data, and 78% on Test data which is higher than other models.

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

Please explain and summarise the various steps performed in this project. There should be proper business interpretation and actionable insights present.

Ans:

- As per the EDA, we can state that, Wife education, Standard of living index, Media exposure are the few variables that plays a vital role in the usage of conceptive method.
- We can also state that wives who are working have a smaller number of children.
- Most of them fall under high or very high standard of living index.
- There are about 7.82% of wives who are still not exposed to media.
- About 44.08% of wives still do not use conceptive methods.
- Educating wives, creating awareness and increase in media exposure are the few methods to increase the usage of conceptive methods. So, we can say that, use of contraceptive method of choice can also be based on their demographic and socio-economic characteristics.
- CART or decision tree would be the best model when compared with the other models.
- Based on the predictions of the model, it would be better to remove few variables of less importance from the model in order to increase performance of the model.
- Also, based on the importance of the model, the primary focuses should be on the variables such as ‘Wife age’, ‘No. of children’, ‘Husband occupation’ and ‘Wife education’.
- We can state that, use of contraceptive method of choice to some extent would be based on their demographic and socio-economic characteristics.

----------THE END----------