Predictive Modeling

GRADED PROJECT

PGP – DSBA

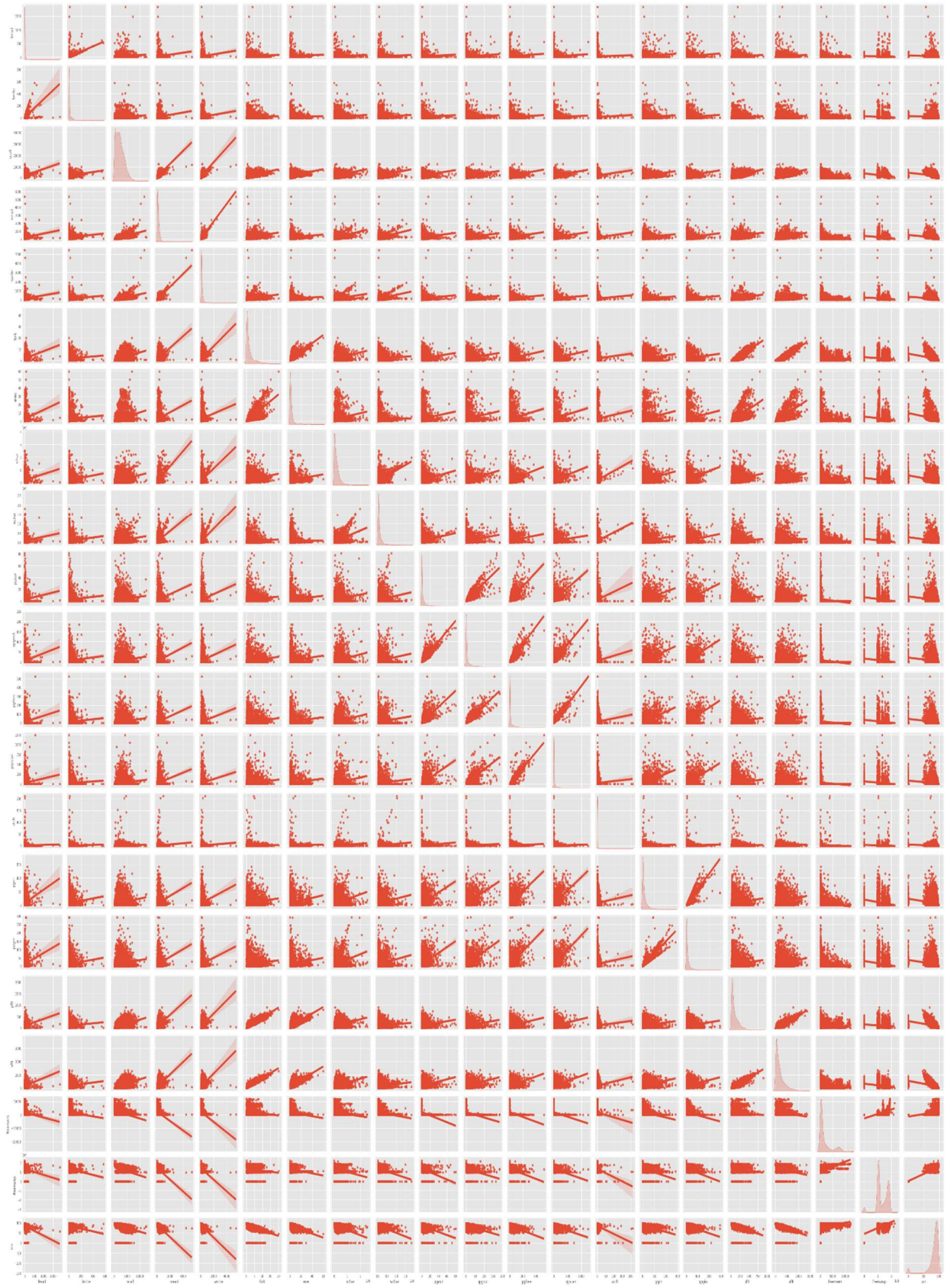
Dated:
08/01/23

Vikash Kumar

Table of Contents

Contents

Table of Contents 2
Linear Regression: Comp-active Data 7
1. Read the data and perform basic analysis such as printing a few rows (head), info, Shape, 5 point data summary, etc. 7
1A Univariate Analysis. 9
Bivariate Analysis 9
Bivariate analysis is stated to be an analysis of any concurrent relation between two variables or attributes. This study explores the relationship of two variables as well as the depth of this relationship to figure out if there are any discrepancies between two variables and any causes of this difference. 9



.....10

Although we can say that there exists some relationship between variables, we can't tell quantify that from the above graphs. So, let's quantify the amounts of correlation using Pearson Correlation and verify the above observations All types of relationship can be observed among variables. Most of variable have negative relationship with usr,freeswap and freemem, while remains are seems in positive relationship.....10

Multivariate Analysis.....11

Null Value Count along with Zero value(Zero treated as Null) 11

1Checking if there are any outliers, and treating outliers is necessary for analysis..... 14

1.3 Encoding of categorical variable 15

One hot encoding is one method of **converting data** to prepare it for an algorithm and get better prediction. With one-hot, we convert each categorical value into a new categorical column and assign a binary value of 1 or 0 to those columns. Each integer value is represented as a binary vector. All the values are zero, and the index is marked with a 1. 15

lread	lwrite	sread	exec	rchar	wchar	pgfree	atch	pgin	freemem	freeswap	usr	_Not_CPU_Bound	
0	1	0	79	0.2	40671	53995	0	0	1.6	4670	1730946	95	0
1	0	0	18	0.2	448	8385	0	0	0	7278	1869002	97	1
2	15	3	159	2.4	125473.5	31950	0	1.2	6	702	1021237	87	1
3	0	0	12	0.2	125473.5	8670	0	0	0.2	7248	1863704	98	1
4	5	1	39	0.4	125473.5	12185	0	0	1	633	1760253	90	1

15

Checking for first 5 data for Training data													
const	lread	lwrite	sread	exec	rchar	wchar	pgfree	atch	pgin	freemem	freeswap	runqsz_Not_CPU_Bound	
694	1	1	1	223	0.6	198703	293578	23.4	2.6	3.8	121	1375446	0
5535	1	1	1	87	0.2	7163	24842	0	0	1.6	1476	1021541	1
4244	1	49	71	225	0.4	83246	53705	7.19	2.79	3.99	82	18	0
2472	1	13	8	300	3	96009	70467	0	0	2.8	772	993909	0
7052	1	17	23	13	1.6	17132	12514	0	0	0	4179	1821682	1

15

Notes: [1] Standard Errors assume that the covariance matrix of the errors is correctly specified. [2] The condition number is large, 6.7e+06. This might indicate that there are strong multicollinearity or other numerical problems. 20

Creating scatterplot with regression line 21

R2 Score comparison among the different models of regression 22

Multivariate Analysis 31

2.Making the data split into train and test 70:30 ratio as per instruction 33

The shape of data of training and test 33

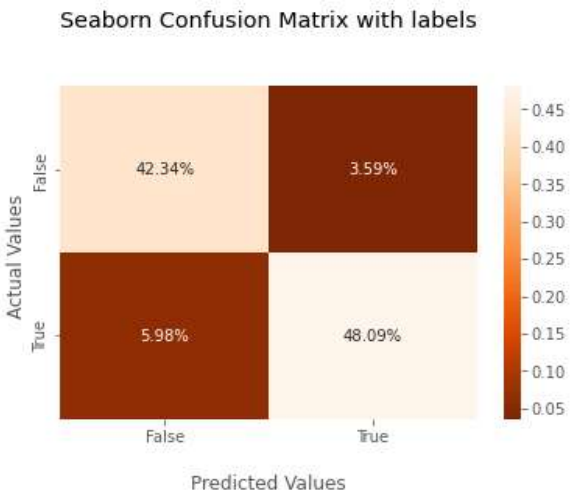
Model Evaluation 42

Confusion Matrix for the test data 44

Model Comparison & Summary 45

Model Comparision					
Model		**Precision**	**Recall**	**f1-score**	**Accuracy**
Logistic R	0	0.88	0.92	0.43	**0.64**
	1	0.93	0.89	0.91	
LDA	0	0.65	0.47	0.55	**0.64**
	1	0.64	0.78	0.7	
CART	0	0.61	0.34	0.43	**0.59**
	1	0.59	0.81	0.68	

..... 45


Seaborn Confusion Matrix with labels
42.34%
3.59%
5.98%
48.09%
False
True
Actual Values
False
True
Predicted Values
0.45
0.40
0.35
0.30
0.25
0.20
0.15
0.10
0.05

.....45

Linear Regression: Comp-active Data

Problem Statement:

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

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

1. **Read the data and perform basic analysis such as printing a few rows (head), info, Shape, 5 point data summary, etc.**

   The data was imported and basic analysis was done. Here are a few snippets:

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

Table 1: The head of the table (first 5 rows)

lread	lwrite	sread	exec	rchar	wchar	pgfree	atch	pgin	freemem	freeswap	runqsz	Not_CPU_Bound
8187	16	12	360	5.81	405250.0	85282.0	43.69	0.6	35.87	387	986647	0
8188	4	0	170	1.80	89489.0	41764.0	4.80	0.8	3.80	263	1055742	1
8189	16	5	289	0.60	325948.0	52640.0	0.60	0.4	28.40	400	969106	1
8190	32	45	254	1.20	62571.0	29505.0	13.03	0.4	23.05	141	1022458	0
8191	2	0	55	4.80	111111.0	22256.0	0.00	0.2	3.40	659	1756514	0

Table 2: The tail of the table (last 5 rows)

Let's have a look at the data types of the columns, and other basic info.

1.The dataset has spreat in 8192 entries with 22 variable columns.

2.The dataset containing THREE Data types of variables, Int64 8 in numbers, float is maximum ie 13 in count whereas only 01 object is there.

3.In very first look the missing data are seems in 'rchar' and 'wchar' columns which also peeping through top 5 value analysis.

```
<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
```

Table 3: Basic Info

Looking at the descriptive features of the dataset.

	count	mean	std	min	25%	50%	75%	max
lread	8192	1.96E+01	53.353799	0	2	7	20	1845
lwrite	8192	1.31E+01	29.891726	0	0	1	10	575
sread	8192	2.10E+02	198.980146	6	86	166	279	5318
exec	8192	2.79E+00	5.212456	0	0.2	1.2	2.8	59.56
rchar	8192	1.96E+05	238446.012	278	34860.5	125473.5	265394.75	2526649
wchar	8192	9.58E+04	140728.464	1498	22977.75	46619	106037	1801623
pgfree	8192	1.19E+01	32.36352	0	0	0	5	523
atch	8192	1.13E+00	5.708347	0	0	0	0.6	211.58
pgin	8192	8.28E+00	13.874978	0	0.6	2.8	9.765	141.2
freemem	8192	1.76E+03	2482.10451	55	231	579	2002.25	12027
freeswap	8192	1.33E+06	422019.427	2	1042623.5	1289289.5	1730379.5	2243187
runqsz_Not_CPU_Bound	8192	5.29E-01	0.499207	0	0	1	1	1

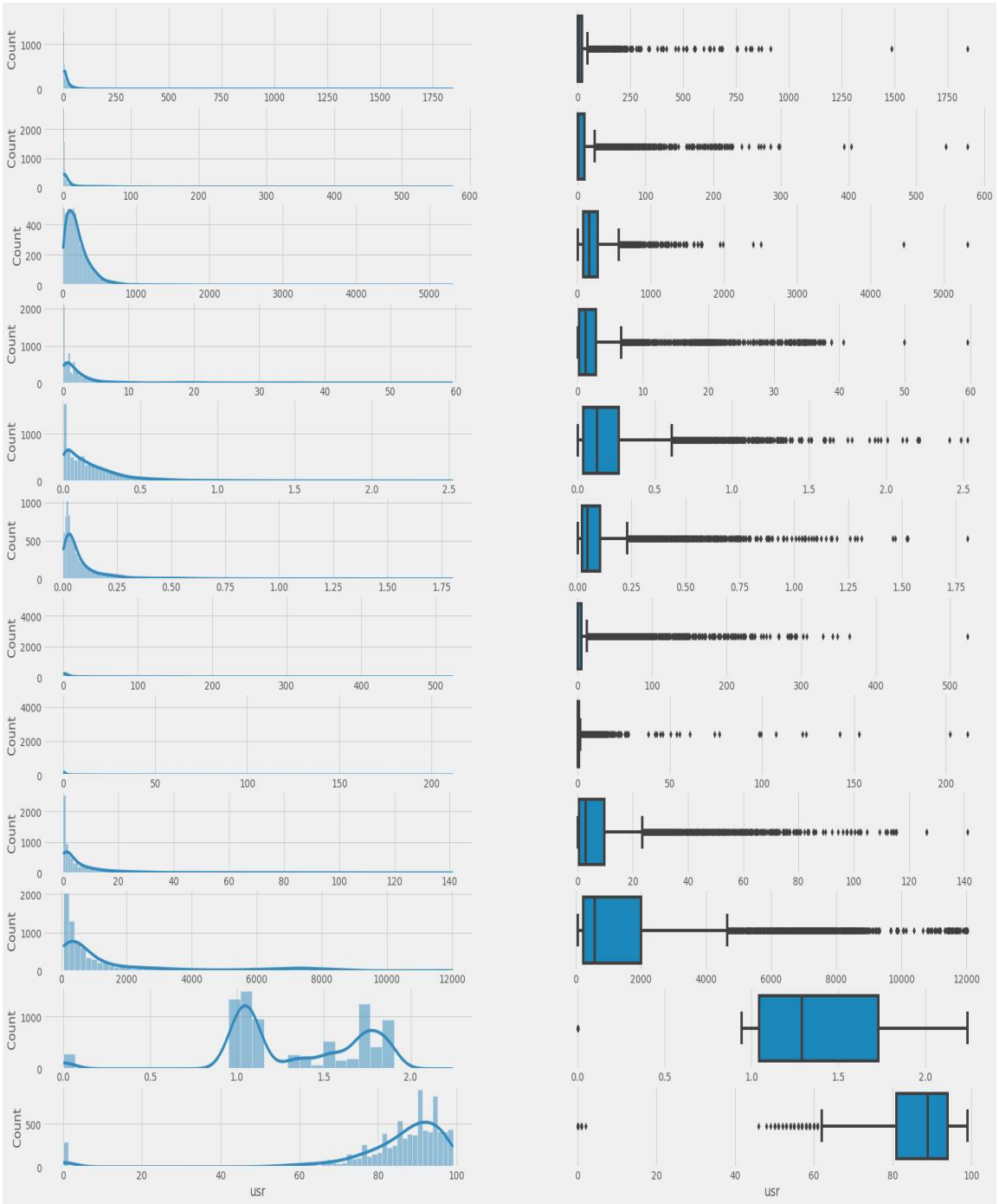
Table 4: Descriptive Features

Except the 'usr' all cases having Mean are higher than the Median hence data is seems Right skewed.
which further will reveal clear after visualization individually.

In most of variable the data giving idea of presence of outliers because the gap in mean and max value is very high.

1A Univariate Analysis.

Univariate analysis is the simplest kind of data analysis in the field of statistics.
This could be either descriptive or inferential in nature. The key thing about the univariate analysis is that there is only one data involved here. While the univariate analysis may be easy to analyze and also is not complex

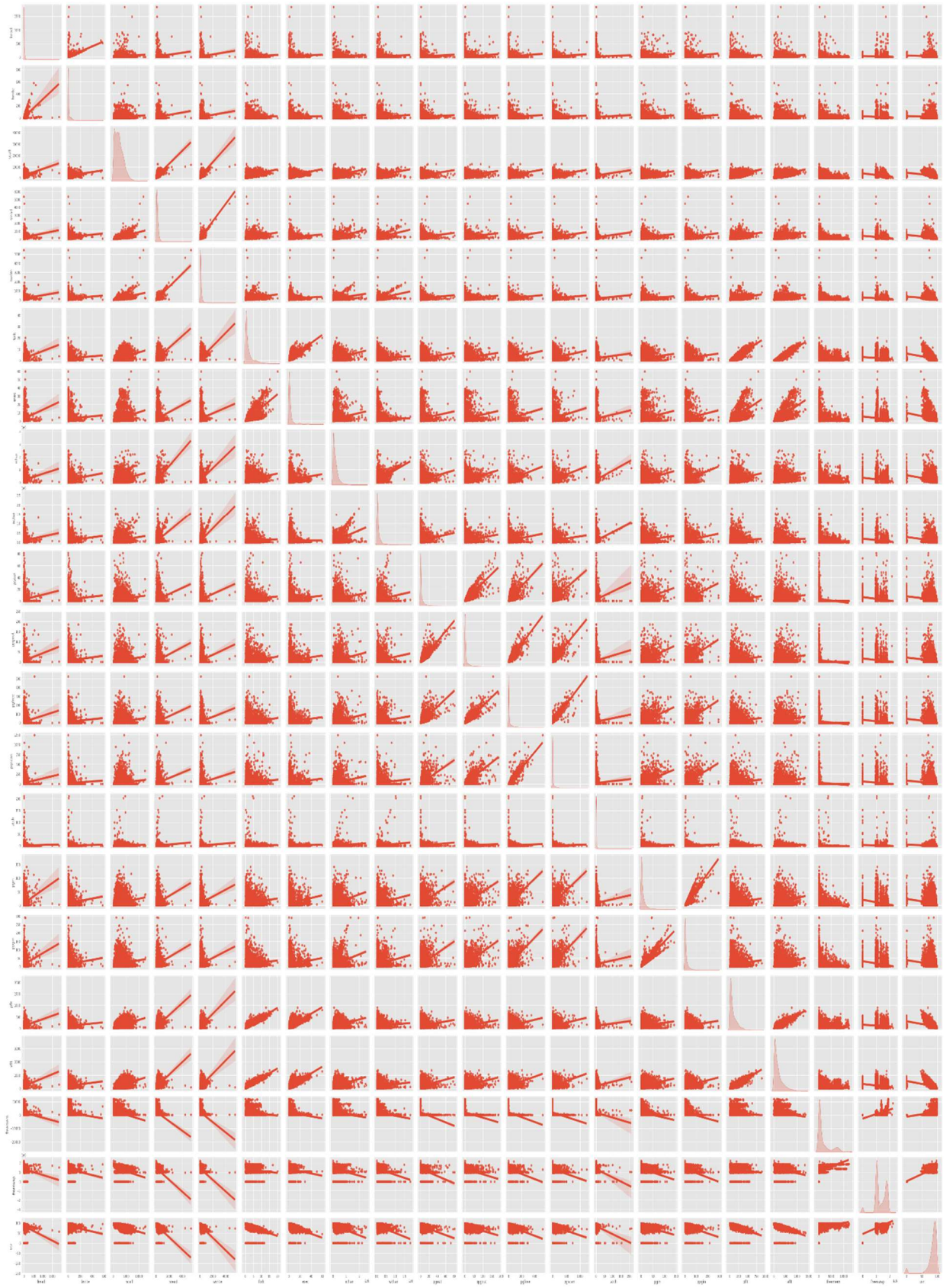


Measurement of Central Tendecy: The data set is following a trend of right skewness except in usr which is left skewed with presence of the large number of outliers

Measurment of Dispersion : The variables like fork,pgout,ppgout,pgfree,pgscan,atch has NO datapoint till 50% ,Where as The variables like lwrite and exec up to 25% there are No point data.While In variable like lread,lwrite,fork,exec,pgout,ppgout,pgfree,pgscan,atch, pgin,ppgin,pflt,usr the minimum numbers are zero.

Bivariate Analysis

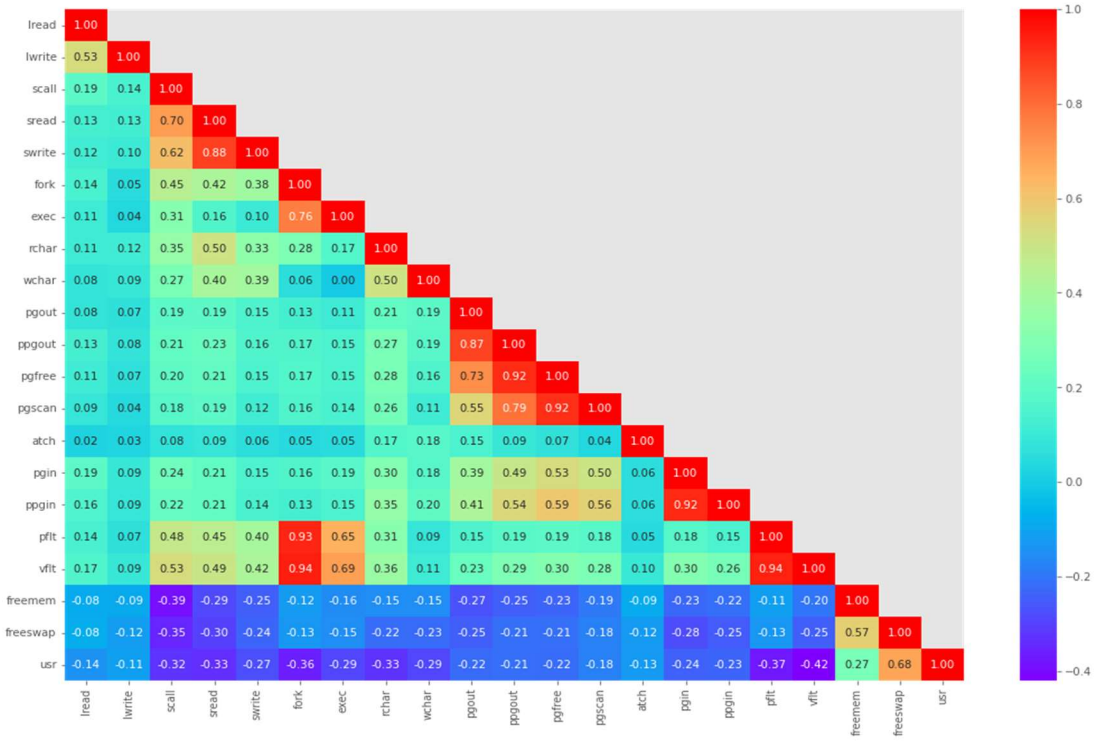
Bivariate analysis is stated to be an analysis of any concurrent relation between two variables or attributes. This study explores the relationship of two variables as well as the depth of this relationship to figure out if there are any discrepancies between two variables and any causes of this difference.



Although we can say that there exists some relationship between variables, we can't tell quantify that from the above graphs. So, let's quantify the amounts of correlation using Pearson Correlation and verify the above observations All types of relationship can be observed among variables. Most of variable have negative relationship with usr,freeswap and freemem, while remains are seems in positive relationship.

Multivariate Analysis

Multivariate analysis encompasses all statistical techniques that are used to analyze more than two variables at once. The aim is to find patterns and correlations between several variables simultaneously—allowing for a much deeper, more complex understanding of a given scenario than you'll get with bivariate analysis.



There is large correlation are found in data set among variables. We are considering high correlation those variables are having more than 0.5% of relationship. While some variables are having very hard relationship among them in more than 0.79% to 0.94%. 0.03% is minimum positive correlation while -0.39% is lowest among all variables. So the overall data variables are under strong relationship.

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.

Null Value Count along with Zero value(Zero treated as Null)

lread: 675: 8.24%
lwrite: 2684: 32.76%
scall: 0: 0.0%
sread: 0: 0.0%
swrite: 0: 0.0%
fork: 21: 0.26%
exec: 21: 0.26%
rchar: 0: 0.0%
wchar: 0: 0.0%
pgout: 4878: 59.55%
ppgout: 4878: 59.55%
pgfree: 4869: 59.44%
pgscan: 6448: 78.71%
atch: 4575: 55.85%
pgin: 1220: 14.89%
ppgin: 1220: 14.89%
pflt: 3: 0.04%
vflt: 0: 0.0%
runqsz: 0: 0.0%

freemem: 0: 0.0%
freeswap: 0: 0.0%
usr: 283: 3.45%

Table 5: Null Values Count

Large number of Zero value are present in data variables. Whereas its highest inpgscan 6448 out of 8192 ie 78.7%. Only 7 variables are there which has NIL zero values. The Zero values are seeming fine but the final decision on its relevance ban be better revealed in consultation with concerned vertical expert.

The missing values are from Number of characters transferred per second by system read calls(104) and Number of characters transferred per second by system write calls(15),which seems not by fault but have some authenticity. So will Impute this.

lread: 0: 0.0%
lwrite: 0: 0.0%
scall: 0: 0.0%
sread: 0: 0.0%
swrite: 0: 0.0%
fork: 0: 0.0%
exec: 0: 0.0%
rchar: 104: 1.27%
wchar: 15: 0.18%
pgout: 0: 0.0%
ppgout: 0: 0.0%
pgfree: 0: 0.0%
pgscan: 0: 0.0%
atch: 0: 0.0%
pgin: 0: 0.0%
ppgin: 0: 0.0%
pflt: 0: 0.0%
vflt: 0: 0.0%
runqsz: 0: 0.0%
freemem: 0: 0.0%
freeswap: 0: 0.0%
usr: 0: 0.0%

Table 6: Null Values Count only for actual Nulls

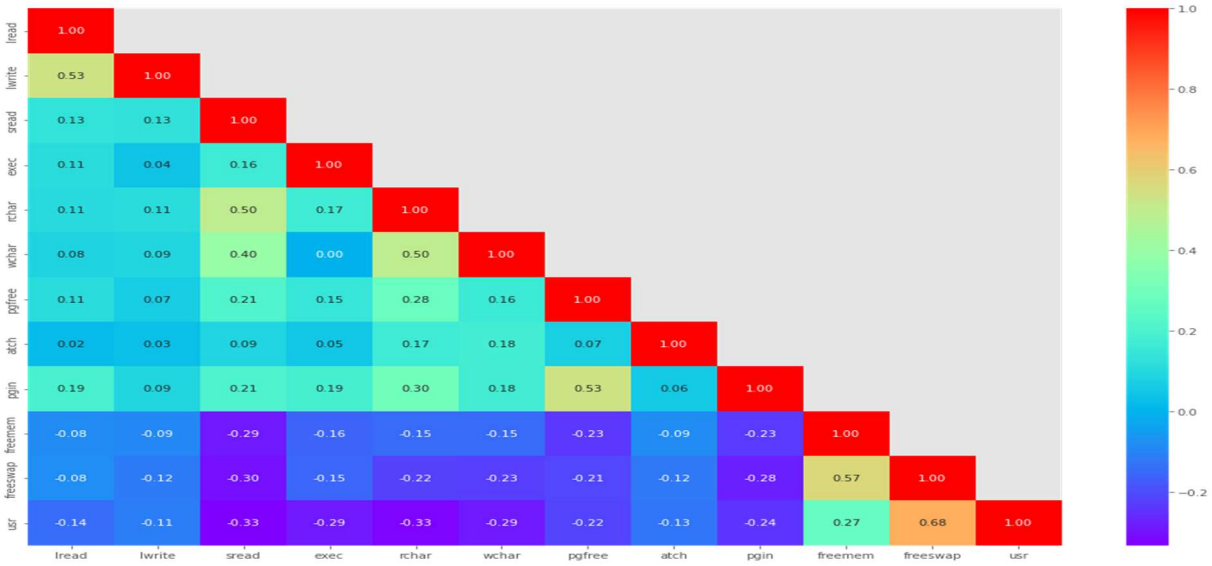
```
lread       0
lwrite      0
scall       0
sread       0
swrite      0
fork        0
exec        0
rchar       0
wchar       0
pgout       0
ppgout      0
pgfree      0
pgscan      0
atch        0
pgin        0
ppgin       0
pflt        0
vflt        0
runqsz      0
freemem     0
freeswap    0
usr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

There are no more null values after treatment.

Possibility of creating new features to treat missing value as zero

We can check the MULTICOLONARITY of data and will see the result in light of Zero values also. The large number of values are present in large number of variables (except 7 nos).If we find the strong correlation than we will go for dropping of variable for further analysis.

Further we are going to drop ['vflt','pgout','ppgout', 'pgscan', 'ppgin','swrite','scall','fork', 'pflt'] the columns which show the high level of correlation. Which will also helpful to get rid of maximum number of Zero values.



.

After the dropping of above mentioned columns the final data with zero figures are:

```
lread: 675: 8.24%
lwrite: 2684: 32.76%
sread: 0: 0.0%
exec: 21: 0.26%
rchar: 0: 0.0%
wchar: 0: 0.0%
pgfree: 4869: 59.44%
atch: 4575: 55.85%
pgin: 1220: 14.89%
runqsz: 0: 0.0%
freemem: 0: 0.0%
freeswap: 0: 0.0%
usr: 283: 3.45%
```

We will carry these Zero values as it is for further analysis because without consultation of business it would be better to carry for analysis.

Checking Duplicated Values

```
: The number of duplicated values are : 0
 Which means data set are free from duplicated values.
```

1Checking if there are any outliers, and treating outliers is necessary for analysis.

Checking for the outliers:

Let's have a quick look at the boxplots for numerical fields:

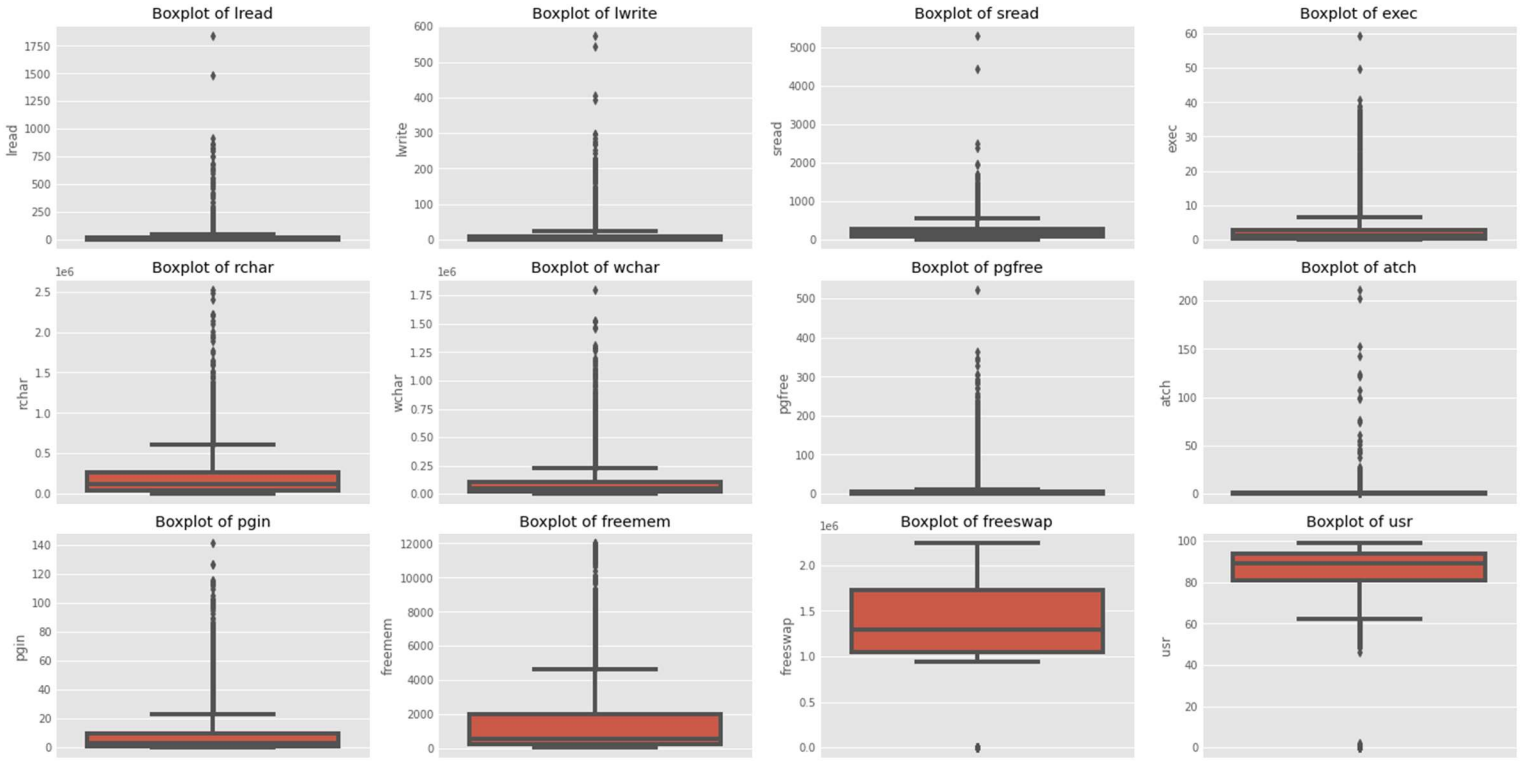


Fig 1: Boxplot of the numeric fields

As seen in the above figure, there are outliers present in almost all the fields. We see outliers present.

Right from beginning the variables were showing the sign of Outliers which are now well visible by box plot method. We have tom perform the linear regression analysis so the treatment of outliers are required.

Outlier Treatment Method:

To treat the outliers in the data, we can create a user defined function (UDF) to calculate the lower range and upper range of the data in a particular column. Once detected, the values below lower range can be replaced by the lower range value, and the values above the upper range can be replaced by the upper range value, of the specific column. This will ensure that the data does not have outliers. The outliers can also be eliminated from the dataset, but this will depend on the share of the outliers in the data, and also a discussion with the business team. In this case, we will proceed with the former approach i.e., replacing the outlier values by the lower range and upper range values, however applicable.

For ease of action, we copy the numerical fields in a new dataset and use our UDFs for detecting and treating outliers. Once done, we plot the boxplot again to see if it worked.

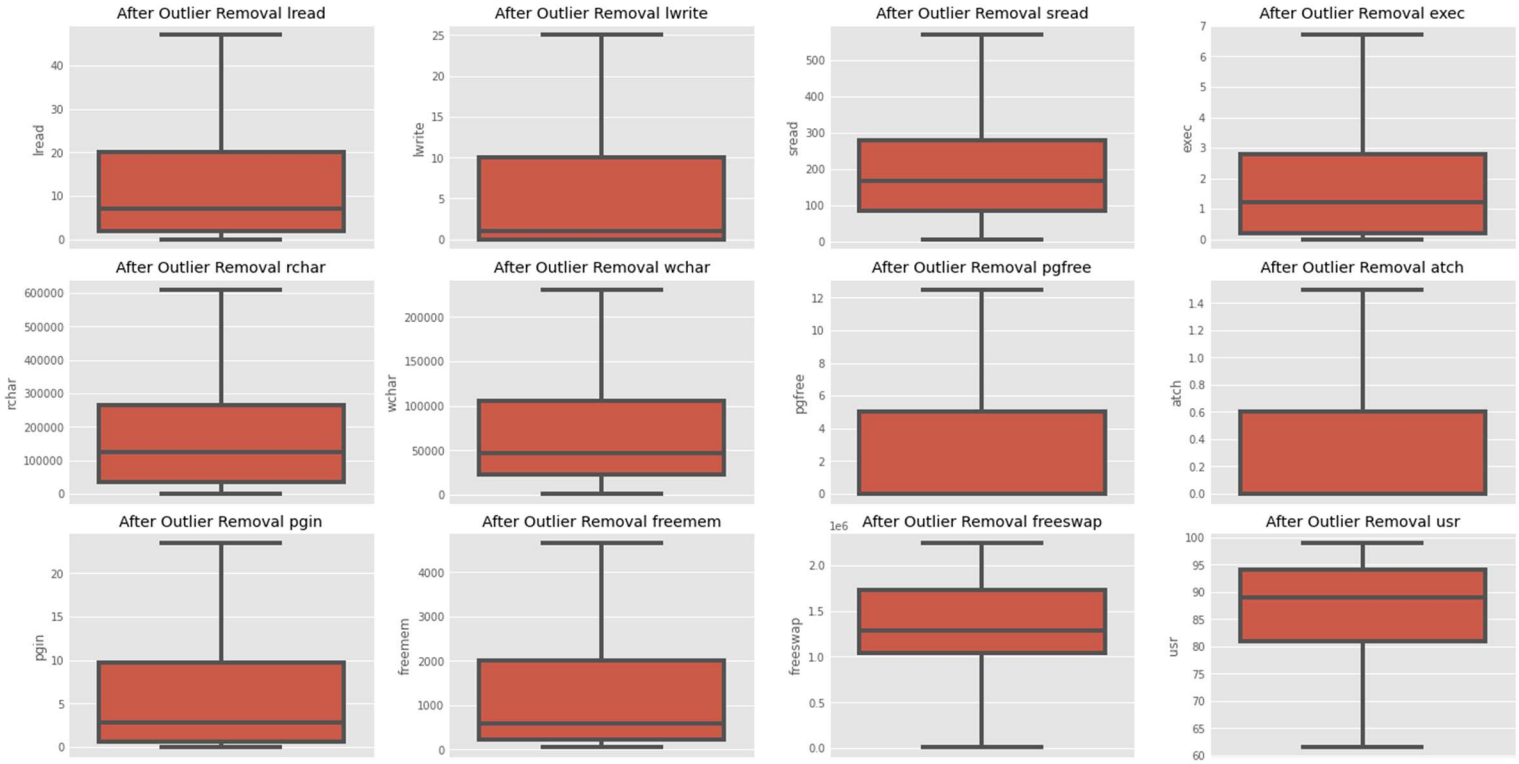


Fig 2: Boxplot of Numeric fields after outlier treatment

As we can see, all the outliers have been treated, and our dataset is ready for further.

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 of categorical variable

Encoding the data by Create Dummy Variables

One hot encoding is one method of **converting data** to prepare it for an algorithm and get better prediction. With one-hot, we convert each categorical value into a new categorical column and assign a binary value of 1 or 0 to those columns. Each integer value is represented as a binary vector. All the values are zero, and the index is marked with a 1.

lread	lwrite	sread	exec	rchar	wchar	pgfree	atch	pgin	freemem	freeswap	usr	_Not_CPU_Bound	
0	1	0	79	0.2	40671	53995	0	0	1.6	4670	1730946	95	0
1	0	0	18	0.2	448	8385	0	0	0	7278	1869002	97	1
2	15	3	159	2.4	125473.5	31950	0	1.2	6	702	1021237	87	1
3	0	0	12	0.2	125473.5	8670	0	0	0.2	7248	1863704	98	1
4	5	1	39	0.4	125473.5	12185	0	0	1	633	1760253	90	1

Splitting data into training and test into 70:30 ratio

```
shape of train: (5734, 13) (5734,)
shape of test:  (2458, 13) (2458,)
```

Checking for first 5 data for Training data													
const	lread	lwrite	sread	exec	rchar	wchar	pgfree	atch	pgin	freemem	freeswap	runqsz_Not_CPU_Bound	
694	1	1	1	223	0.6	198703	293578	23.4	2.6	3.8	121	1375446	0
5535	1	1	1	87	0.2	7163	24842	0	0	1.6	1476	1021541	1
4244	1	49	71	225	0.4	83246	53705	7.19	2.79	3.99	82	18	0
2472	1	13	8	300	3	96009	70467	0	0	2.8	772	993909	0
7052	1	17	23	13	1.6	17132	12514	0	0	0	4179	1821682	1

Checking for first 5 data for Test data													
const	lread	lwrite	sread	exec	rchar	wchar	pgfree	atch	pgin	freemem	freeswap	runqsz_Not_CPU_Bound	
3894	1	27	39	53	0.2	26592	54394	0	0	0.4	7762	1875466	1
4276	1	1	0	85	0.4	16667	36431	0	0	1	2979	1010114	1
3414	1	9	7	247	0.4	14513	61905	30.4	10.4	14.8	89	11	0
4165	1	32	4	182	5.6	337517	94832	1	1.4	4.6	1300	1535309	0
7385	1	16	3	259	1.4	73537	237547	0	0	5.6	2114	988600	0

Table 7: Top five data of train and test data

Checking the dimensions of the training and test data

```
X_train (5734, 13)
X_test (2458, 13)
y_train (5734,)
y_test (2458,)
Total Obs 8192
```

 # Let us explore the coefficients for each of the independent attributes

```
The coefficient for const is 0.0
The coefficient for lread is -0.021770426250442922
The coefficient for lwrite is 0.011991026779689886
The coefficient for sread is -0.004635783924150994
The coefficient for exec is -0.6590513352461455
The coefficient for rchar is -5.32436563818823e-06
The coefficient for wchar is -1.0327712182192639e-05
The coefficient for pgfree is -0.0388663412523446
The coefficient for atch is -0.06403975116432621
The coefficient for pgin is 0.057328201948089415
The coefficient for freemem is -0.001823048783988082
The coefficient for freeswap is 3.083293653151787e-05
The coefficient for runqsz_Not_CPU_Bound is 7.678482792501638
```

The negative regression estimation coefficient is interpreted as the two variables tested having opposite associations. Referring to the case study example above, the negative estimated selling price coefficient indicates that
the selling price (X) variable affects the bread sales variable (Y) in a negative direction

The intercept for our model is 47.29879439245825
The coefficient of determination R^2 of the prediction on Train set
0.6067705563673178
The coefficient of determination R^2 of the prediction on Test set
0.5944574745330722
The Root Mean Square Error (RMSE) of the model is for testing set is
12.158125825519342

Check Multi-collinearity using VIF (States Model)

```
const ---> 21.327010675421555
lread ---> 1.4546044692142015
lwrite ---> 1.4197924758091527
sread ---> 1.5101083050920125
exec ---> 1.0960449836534474
rchar ---> 1.6957841057160048
wchar ---> 1.4518010919102635
pgfree ---> 1.4555576899765845
atch ---> 1.0517465662176666
pgin ---> 1.5354544063851836
freemem ---> 1.5916788887512348
freeswap ---> 1.6188621116139021
runqsz_Not_CPU_Bound ---> 1.0763928904314977
```

On basis of VIF factors all variable having inflation factor is in tune of 1.45 to 1.69 which is permissible limit hence doesn't show any multicoll inearity among variables.

Model Number 1 :Using Statsmodels OLS

OLS Regression Results

Dep. Variable:	usr	**R-squared:**	0.607
Model:	OLS	**Adj. R-squared:**	0.606
Method:	Least Squares	**F-statistic:**	735.6
Date:	Sat, 07 Jan 2023	**Prob (F-statistic):**	0.00
Time:	15:15:33	**Log-Likelihood:**	-22063.
No. Observations:	5734	**AIC:**	4.415e+04
Df Residuals:	5721	**BIC:**	4.424e+04
Df Model:	12		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	47.2988	0.707	66.859	0.000	45.912	48.686
lread	-0.0218	0.003	-6.603	0.000	-0.028	-0.015
lwrite	0.0120	0.006	1.906	0.057	-0.000	0.024
sread	-0.0046	0.001	-5.157	0.000	-0.006	-0.003
exec	-0.6591	0.030	-21.977	0.000	-0.718	-0.600
rchar	-5.324e-06	8.02e-07	-6.638	0.000	-6.9e-06	-3.75e-06
wchar	-1.033e-05	1.28e-06	-8.064	0.000	-1.28e-05	-7.82e-06
pgfree	-0.0389	0.006	-6.822	0.000	-0.050	-0.028
atch	-0.0640	0.027	-2.335	0.020	-0.118	-0.010
pgin	0.0573	0.014	4.216	0.000	0.031	0.084
freemem	-0.0018	7.71e-05	-23.656	0.000	-0.002	-0.002
freeswap	3.083e-05	4.59e-07	67.236	0.000	2.99e-05	3.17e-05

runqsz_Not_CPU_Bound	7.6785	0.313	24.550	0.000	7.065	8.292

Omnibus:	1560.795	**Durb**
Prob(Omnibus):	0.000	**Jarqu**
Skew:	-1.404	
Kurtosis:	6.417	

The variation in the independent variable which is explained by the dependent variable is 60.6771 %

Get the Predictions on test set

```
3894     97.474819
4276     79.604101
3414     43.900524
4165     84.442552
7385     68.966800
          ...
4744    101.919644
6918     82.214915
1556     97.834802
1577     91.464231
453      71.533416
Length: 2458, dtype: float64
```

The Root Mean Square Error (RMSE) of the model is for the training set is 11.34608877506499

Model predict on the basis of Train data

```
694      83.100170
5535     83.035689
4244     44.400537
2472     71.903126
7052    102.097053
          ...
7935     76.064201
5192     98.123877
3980     75.235864
235      82.431169
5157    100.760428
Length: 5734, dtype: float64
```

Model Number 1 :Using Statsmodels OLS

OLS Regression Results

Dep. Variable:	usr	**R-squared:**	0.607
Model:	OLS	**Adj. R-squared:**	0.606
Method:	Least Squares	**F-statistic:**	801.8
Date:	Sun, 08 Jan 2023	**Prob (F-statistic):**	0.00
Time:	16:44:16	**Log-Likelihood:**	-22065.
No. Observations:	5734	**AIC:**	4.415e+04
Df Residuals:	5722	**BIC:**	4.423e+04
Df Model:	11		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P>|t|**	**[0.025**	**0.975]**
const	47.4815	0.701	67.726	0.000	46.107	48.856
lread	-0.0186	0.003	-6.539	0.000	-0.024	-0.013
sread	-0.0046	0.001	-5.108	0.000	-0.006	-0.003
exec	-0.6609	0.030	-22.043	0.000	-0.720	-0.602
rchar	-5.278e-06	8.02e-07	-6.582	0.000	-6.85e-06	-3.71e-06
wchar	-1.03e-05	1.28e-06	-8.041	0.000	-1.28e-05	-7.79e-06
pgfree	-0.0387	0.006	-6.794	0.000	-0.050	-0.028
atch	-0.0644	0.027	-2.348	0.019	-0.118	-0.011
pgin	0.0559	0.014	4.117	0.000	0.029	0.083
freemem	-0.0018	7.71e-05	-23.660	0.000	-0.002	-0.002
freeswap	3.077e-05	4.57e-07	67.274	0.000	2.99e-05	3.17e-05
runqsz_Not_CPU_Bound	7.6672	0.313	24.513	0.000	7.054	8.280

Omnibus:	1564.648	**Durbin-Watson:**	2.054

Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	4698.714
Skew:	-1.407	**Prob(JB):**	0.00
Kurtosis:	6.428	**Cond. No.**	6.70e+06

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 6.7e+06. This might indicate that there are
strong multicollinearity or other numerical problems.

Creating scatterplot with regression line

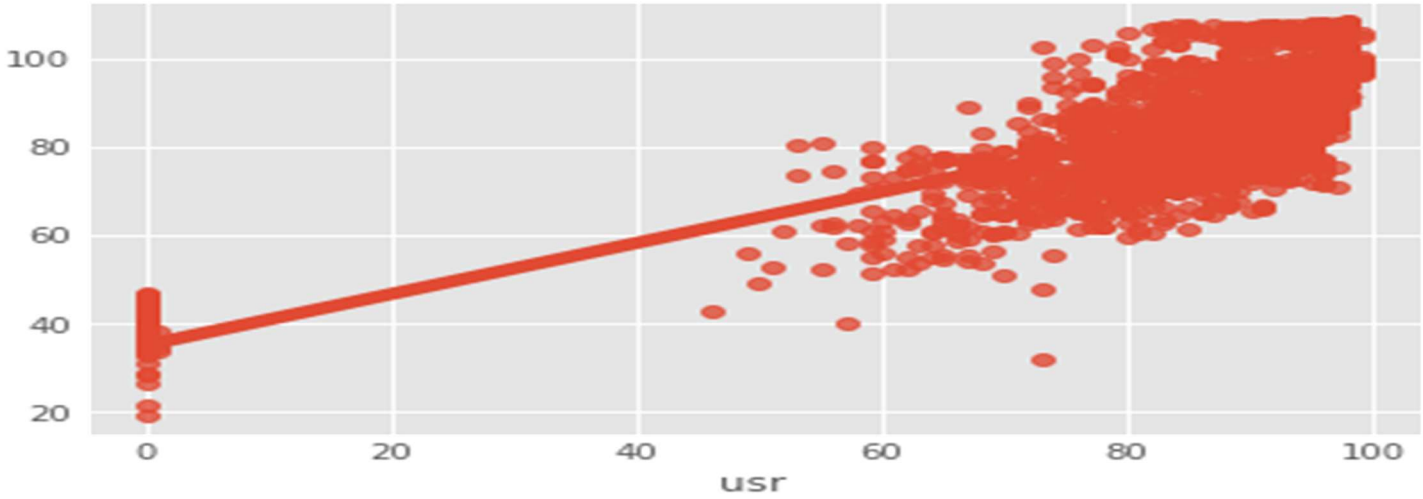
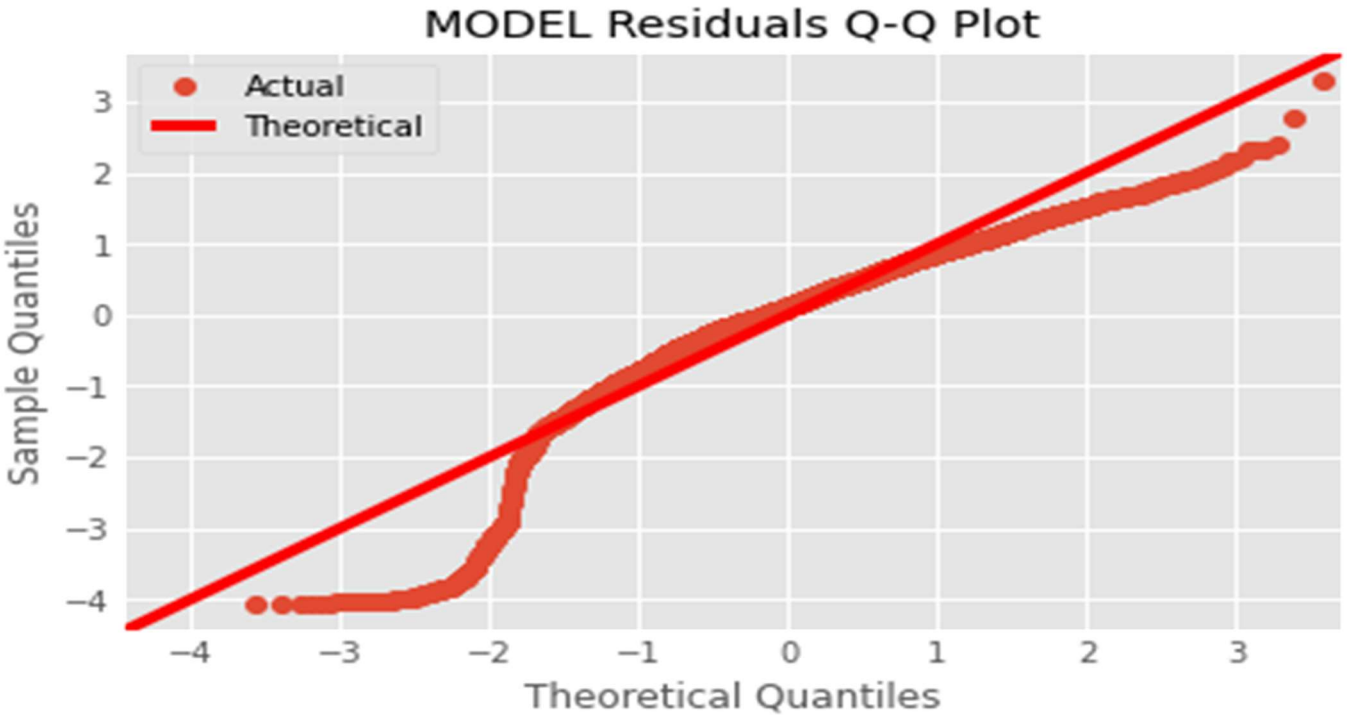


Fig 3: scatterplot

Prediction of linear Regression

```
(47.3) * const + (-0.02) * lread + (0.01) * lwrite + (-0.0) * sread + (-0.66) * exec + (-0
.0) * rchar + (-0.0) * wchar + (-0.04) * pgfree + (-0.06) * atch + (0.06) * pgin + (-0.0)
* freemem + (0.0) * freeswap + (7.68) * runqsz_Not_CPU_Bound +
```

Q-Q plot we will be using scipy's probplot function where we compare a variable of our chosen to a normal probability.



Plot for residual error

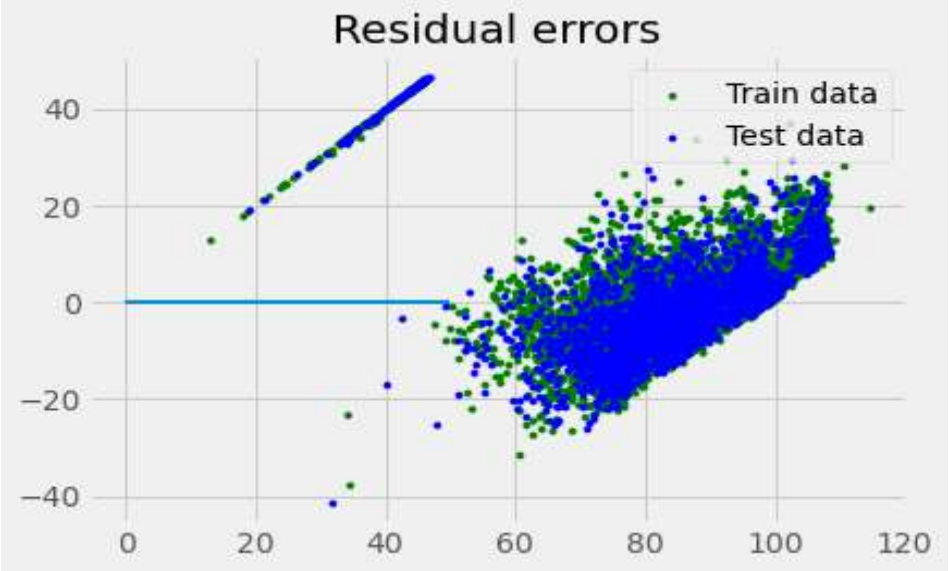
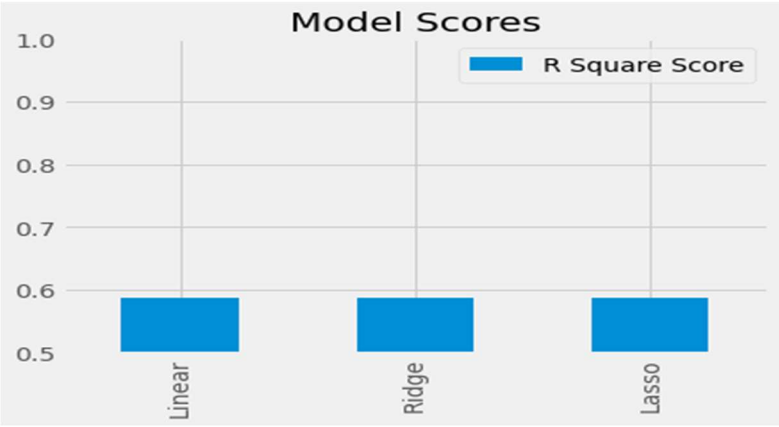


Fig 4: Plot for Residual error

R2 Score comparison among the different models of regression



R Square Score	
Linear	0.587004
Ridge	0.587013
Lasso	0.587004

Table 8: Showing the R Sq comparison in type of regression

The VIf after the generating of model number two.

```
                const ---> 21.327010675421555
                lread ---> 1.4546044692142015
               lwrite ---> 1.4197924758091527
                sread ---> 1.5101083050920125
                 exec ---> 1.0960449836534474
                rchar ---> 1.6957841057160048
                wchar ---> 1.4518010919102635
               pgfree ---> 1.4555576899765845
                 atch ---> 1.0517465662176666
                 pgin ---> 1.5354544063851836
              freemem ---> 1.5916788887512348
             freeswap ---> 1.6188621116139021
runqsz_Not_CPU_Bound ---> 1.0763928904314977
```

Table 9: Data Tail after applying clusters

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

Model Comparison:
There is no majar changes are observed in model number 1 & 2 only the value of F-Statistic is changing while R sq and R sq Adj are not changing. Hence the model are final and no further requirement of model generation. Finally the The variation in the in dependent variable which is explained by the dependent variable is **60.6771 %**

Model summary

R-value: represents the correlation between the dependent and independent variable. A value greater than 0.4 is taken for further analysis. In this case, the value is .713, which is good.

R-square: shows the total variation for the dependent variable that could be explained by the independent variables. A value greater than 0.6 shows that the model is effective enough to determine the relationship. In this case, the value is .607, which is good.

Adjusted R-square: shows the generalization of the results i.e. the variation of the sample results from the population in multiple regression. It is required to have a difference between R-square and Adjusted R-square minimum. In this case, the value is .606, which is not far off from .607, so it is good. Therefore, the model summary table is satisfactory to proceed with the next step. However, if the values were unsatisfactory, then there is a need for adjusting the data until the desired results are obtained.

Anova Table: Elements of this table relevant for interpreting the results are:

P-value/ Sig value: Generally, 95% confidence interval or 5% level of the significance level is chosen for the study. Thus the p-value should be less than 0.05. In the above table, it is .000. Therefore, the result is significant. F-ratio: It represents an improvement in the prediction of the variable by fitting the model after considering the inaccuracy present in the model. A value is greater than 1 for F-ratio yield efficient model. In the above table, the value is 735.6, which is good. These results estimate that as the p-value of the ANOVA table is below the tolerable significance level, thus there is a possibility of rejecting the null hypothesis in further analysis.

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

The given data set the 60.7% data are found in linear relation , on the basis algorithm performed the Portion of time (%) that cpus run in user mode are depending on following variables. These have some positive relation while some has negative relations. There is also variables that have no overall impact on deciding the cpu run mode.
The algorithm is coming out:

```
(47.3) * const + (-0.02) * lread + (0.01) * lwrite + (-0.0) * sread + (-0.66) * exec + (-0
.0) * rchar + (-0.0) * wchar + (-0.04) * pgfree + (-0.06) * atch + (0.06) * pgin + (-0.0)
* freemem + (0.0) * freeswap + (7.68) * runqsz_Not_CPU_Bound +
```

This means the unit increase change in Process run queue size (The number of kernel threads in memory that are waiting for a CPU to run. 7.68 times increases in user mode if other aspects are constant.
Whereas There would be **no impact** of Number of systems read calls per second, Number of characters transferred per second by system read calls, Number of characters transferred per second by system write calls, - Number of memory pages available to user processes, Number of disk blocks available for page swapping.
There is Negative impact of few variables like Reads (transfers per second) between system memory and user memory, Number of system exec calls per second, Number of pages per second placed on the free list.

So the Process run queue size (The number of kernel threads in memory that are waiting for a CPU to run. 7.68 times increases in user mode will affect more.

1.4 Explain and summarize the various steps performed in this project

Linear Regression: *Linear Regression represents some relationship between the dependent variable (y) and the independent variable (X).* It is used to classify the data.

We have implemented Linear Regression in Python in just 7 steps:

Step 1: Imported the required libraries

Here, we have imported Pandas, NumPy, math, statsmodels, matplotlib and sklearn and other libraries to work with.

Step 2: Read the data using Pandas library

We have used pd.read_csv function to read the data.

There we did exploratory data analysis, the Data types, shape, Null Value detection EDA, 5-point summary. Performed Univariate, Bivariate Analysis, Multivariate Analysis, Outlier detection and treatment, checked Multicololinearity.

Then checked for categorical data and changed them to encoded mode for further analysis.

Step 3: Distribute the data into X and Y axis

We have distributed the columns of the data into X and Y axis to visualize and predict the model.

Step 4: Split the data into train and test set

We have splited the data into the ratio of (7:3) as per guideline. That means, training data=70% and test data=30%.

Step 5: Fit the model and make prediction

We'll then fit the model using fit() method and will predict the data to build our model.

Step 6: Visualize the data using matplotlib

We'll visualize the data of the model using matplotlib library to have a better vision

Step 7: Calculated the accuracy of the model

We had calculated and print the accuracy of our model which will tell us how precise our predicted model is. On the basis of R sq, R sq adj, F-Stat, hypothesis testing data, coefficient values and other parameters.

Step by Step for Linear regression

Step 1	seaprating X and y
	X = df.drop('target', axis = 1)
	y = df['target']
Step 2	X_train, X_test, y_train, y_test = train_test_split(X, y, test_size =
Step 3	model = SomeRegressor()/Classifier()
Step 4	model.fit(X_train, y_train)
Step 5	y_pred_train = model.predict(X_train)
	y_pred_test= model.predict(X_test)
Step 6	classification_report(y_train, y_pred_train)
	classification_report(y_test, y_pred_test)

Conclusion

Thus, we can conclude that Linear Regression is a strong tool in Machine Learning which is used to build and analyze the models based on continuous variables representing the relationship between X and y.

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.

Looking for top 5 of data set:

Top 5 data of 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	Primary	Secondary	3	Scientology	No	2	High	Exposed	No
1	45	Uneducated	Secondary	10	Scientology	No	3	Very High	Exposed	No

2	43	Primary	Secondary	7	Scientology	No	3	Very High	Exposed	No
3	42	Secondary	Primary	9	Scientology	No	3	High	Exposed	No
4	36	Secondary	Secondary	8	Scientology	No	3	Low	Exposed	N

Table 10: Top 5 of Dataset for QN2

The data information for analysis of primary aspects.

```
<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
```

Then shape of data is Total 1473 rows are present in data settled 10 columns. All type of data are present like float64(2), int64(1), object (7), uint8(3).

Five point statistical summary of dataset.

	count	unique	top	freq	mean	std	min	25%	50%	75%	max
Wife_age	1402.00	NaN	NaN	NaN	32.61	8.27	16.00	26.00	32.00	39.00	49.00
Wife_ education	1473	4	Tertiary	577	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Husband_education	1473	4	Tertiary	899	NaN	NaN	NaN	NaN	NaN	NaN	NaN
No_of_children_born	1452.00	NaN	NaN	NaN	3.25	2.37	0.00	1.00	3.00	4.00	16.00
Wife_religion	1473	2	Scientology	1253	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Wife_working	1473	2	No	1104	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Husband_Occupation	1473.00	NaN	NaN	NaN	2.14	0.86	1.00	1.00	2.00	3.00	4.00
Standard_of_living_index	1473	4	Very High	684	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Media_exposure	1473	2	Exposed	1364	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Contraceptive_method_used	1473	2	Yes	844	NaN	NaN	NaN	NaN	NaN	NaN	NaN

Checking of Duplicate Values

```
The number of duplicated values are : 80
```

Dropping the Duplicated values

The shape of data after dropping

```
Before (1473, 10)
After (1393, 10)
```

Checking & Imputing of Missing Values

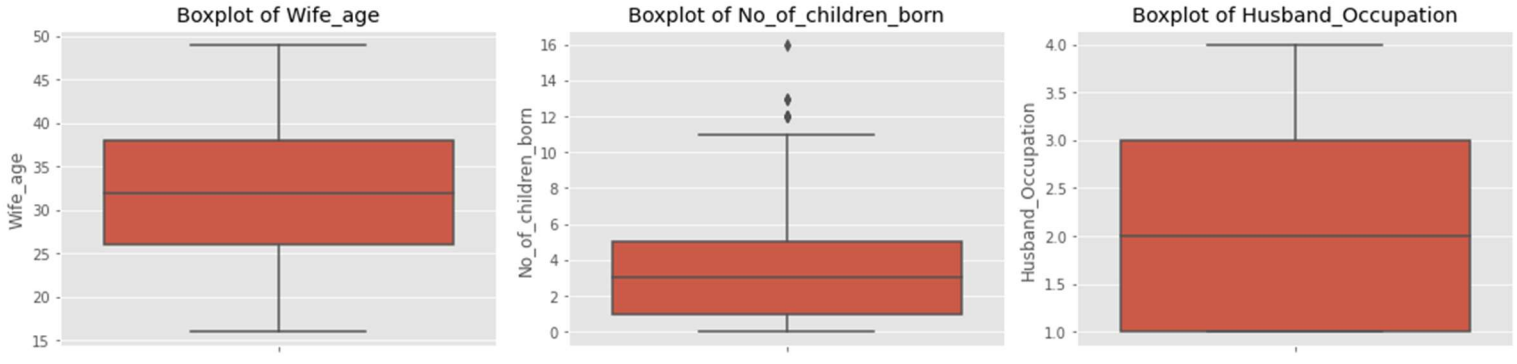
The missing values are:

```
Wife_age: 67: 4.81%
Wife_ education: 0: 0.0%
Husband_education: 0: 0.0%
No_of_children_born: 21: 1.51%
Wife_religion: 0: 0.0%
Wife_working: 0: 0.0%
Husband_Occupation: 0: 0.0%
Standard_of_living_index: 0: 0.0%
Media_exposure: 0: 0.0%
Contraceptive_method_used: 0: 0.0%
```

Imputation of Missing value by “MEDIAN”

```
Wife_age: 0: 0.0%
Wife_ education: 0: 0.0%
Husband_education: 0: 0.0%
No_of_children_born: 0: 0.0%
Wife_religion: 0: 0.0%
Wife_working: 0: 0.0%
Husband_Occupation: 0: 0.0%
Standard_of_living_index: 0: 0.0%
Media_exposure: 0: 0.0%
Contraceptive_method_used: 0: 0.0%
```

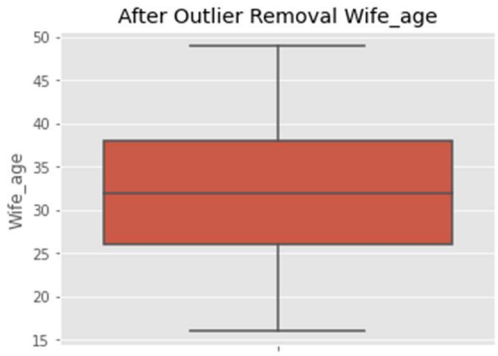
Outlier Detection and Removal

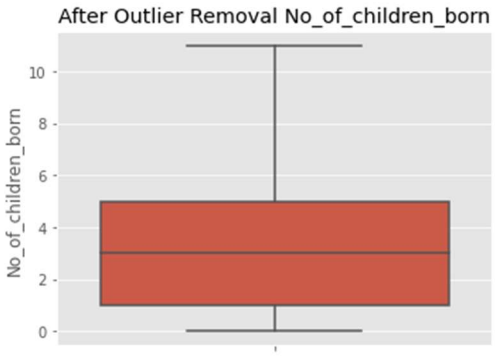


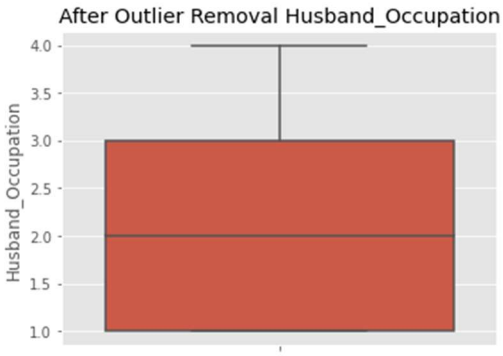
There s presence of outlier in No of children born so we have treat the outlier for further processes.

To treat the outliers in the data, we can create a user defined function (UDF) to calculate the lower range and upper range of the data in a particular column. Once detected, the values below lower range can be replaced by the lower range value, and the values above the upper range can be replaced by the upper range value, of the specific column. This will ensure that the data does not have outliers. The outliers can also be eliminated from the dataset, but this will depend on the share of the outliers in the data, and also a discussion with the business

team. In this case, we will proceed with the former approach i.e., replacing the outlier values by the lower range and upper range values, however applicable.

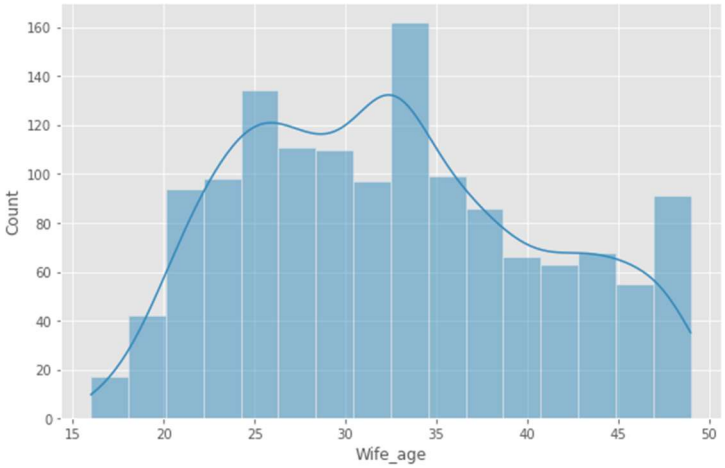


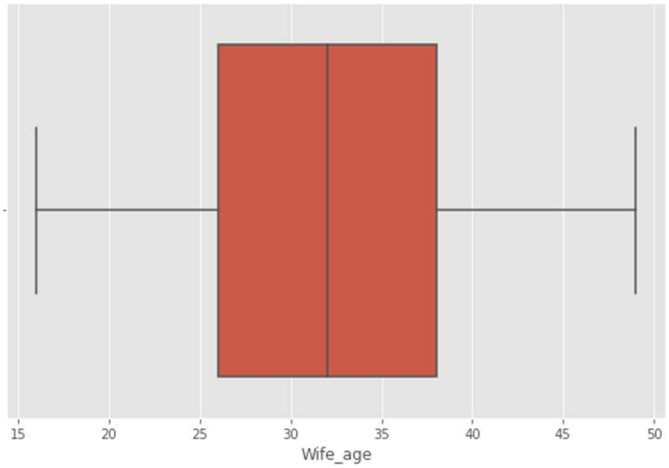


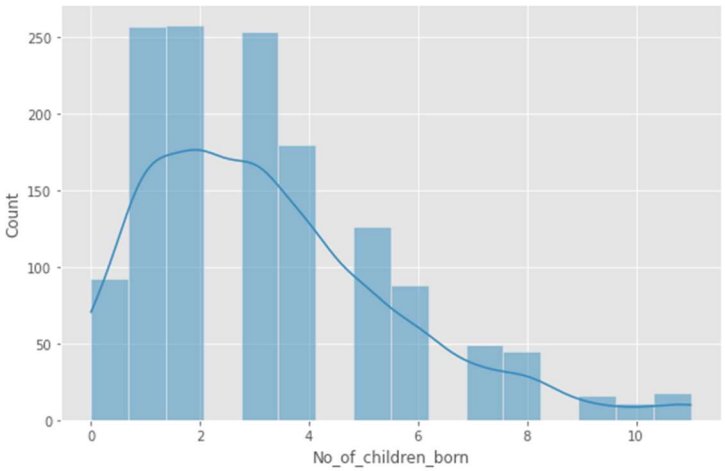


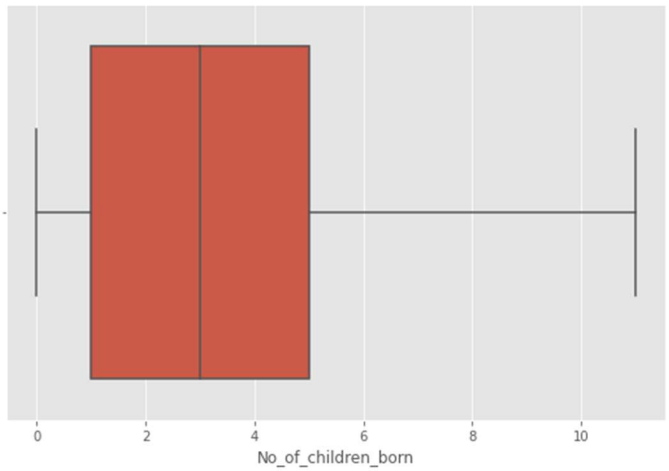
Univariate Analysis

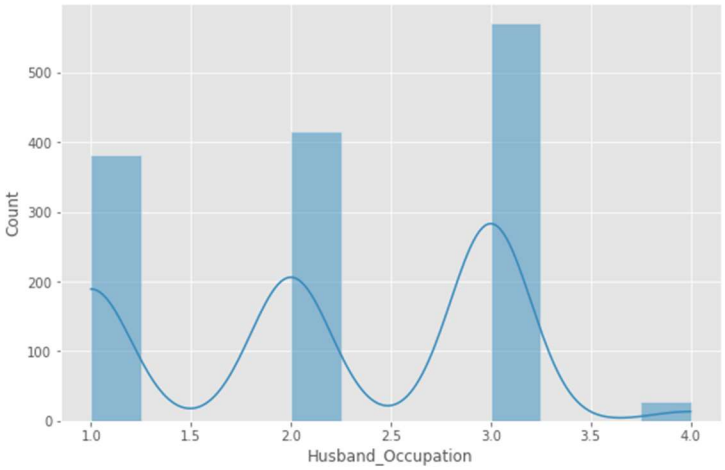
Univariate analysis is the simplest kind of data analysis in the field of statistics. This could be either descriptive or inferential in nature. The key thing about the univariate analysis is that there is only one data involved here. While the univariate analysis may be easy to analyze and also is not complex

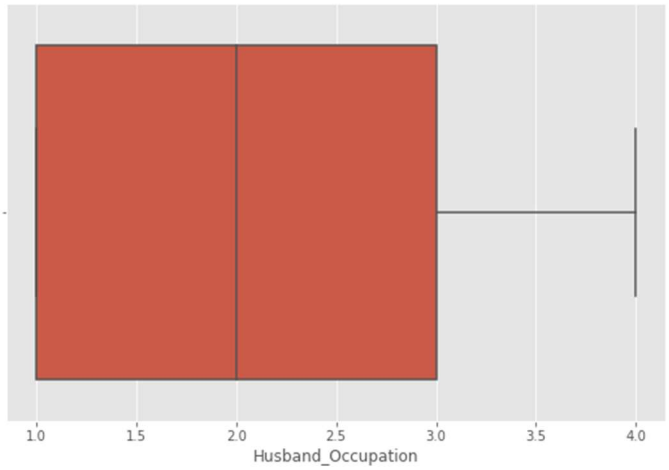

Count
Wife_age

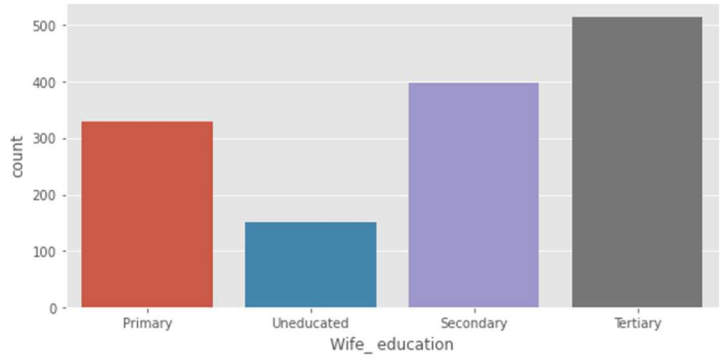

Wife_age

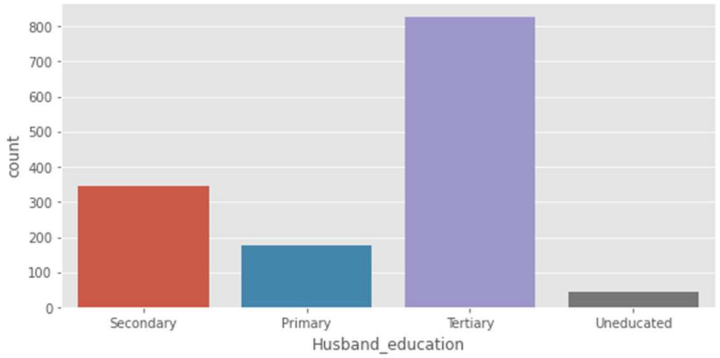

Count
No_of_children_born

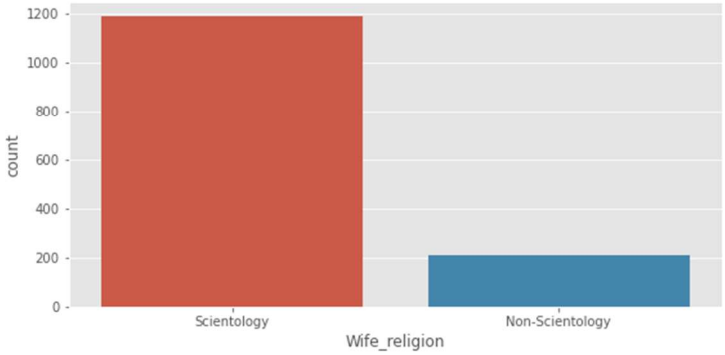

No_of_children_born

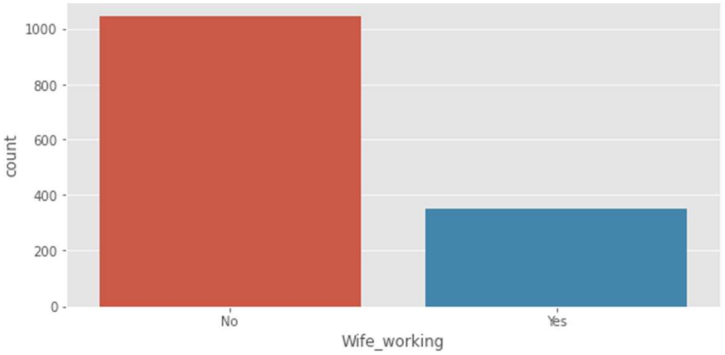

Count
Husband_Occupation

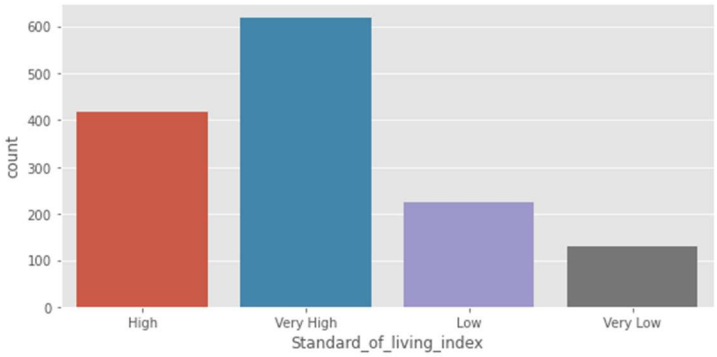

Husband_Occupation

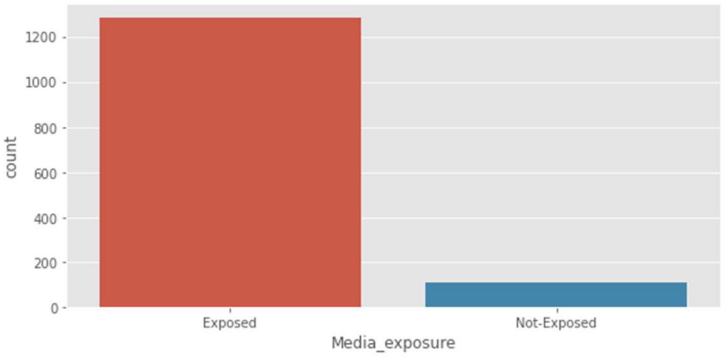


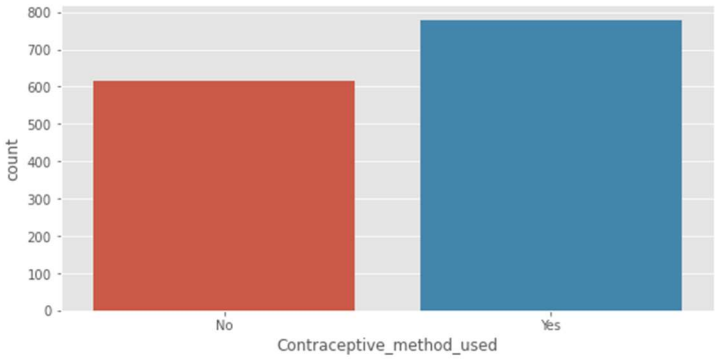


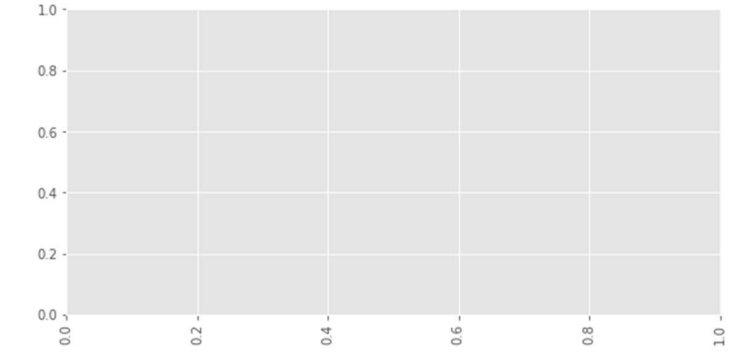








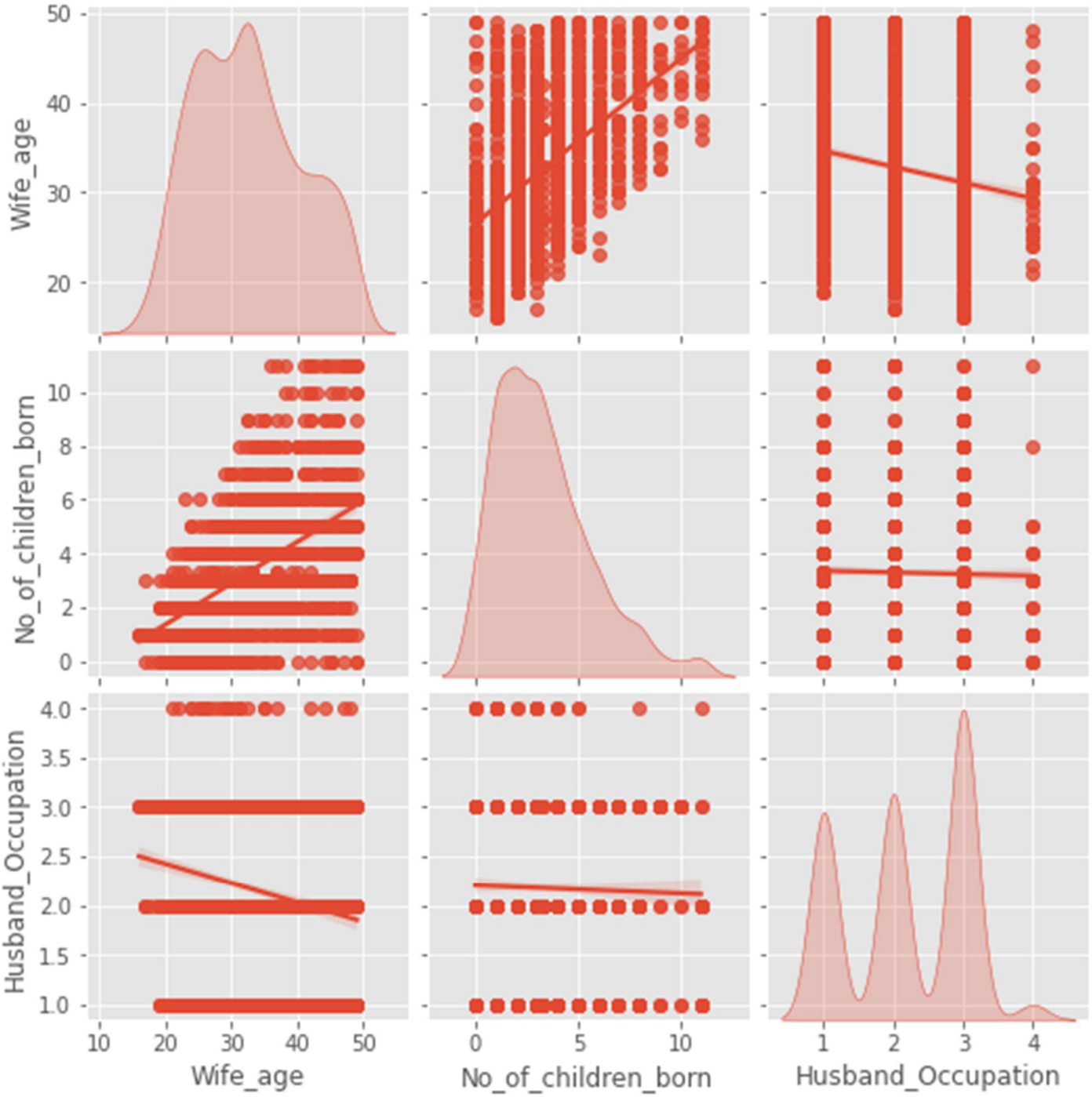




Visualization showing that the proportion of user and Non user of contraceptive are almost equal. Use of contraceptive is almost independent of media exposure and age of wife. In the data the majority are with women which are using contraceptive methods.

Bivariate Analysis

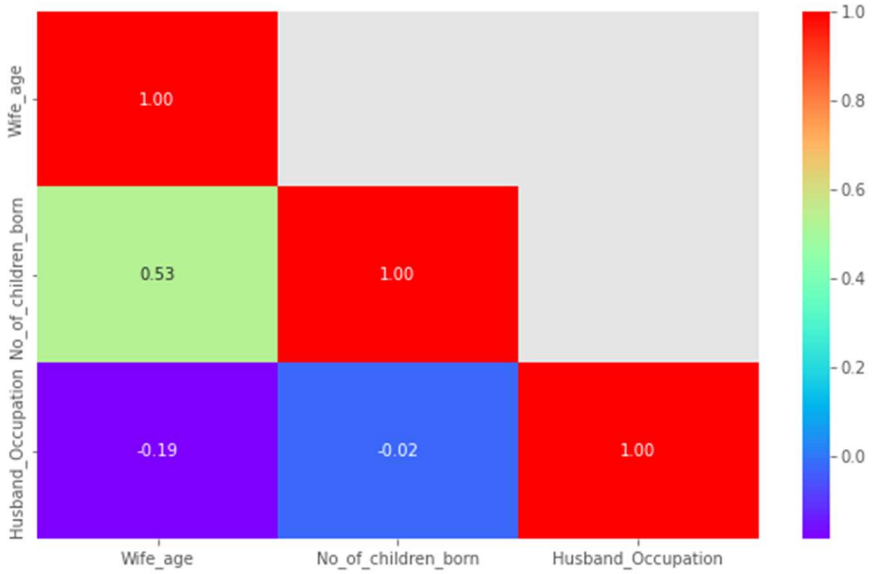
Bivariate analysis is stated to be an analysis of any concurrent relation between two variables or attributes. This study explores the relationship of two variables as well as the depth of this relationship to figure out if there are any discrepancies between two variables and any causes of this difference.



There is No such good trend are visible in bivariate analysis.

Multivariate Analysis

Multivariate analysis encompasses all statistical techniques that are used to analyze more than two variables at once. The aim is to find patterns and correlations between several variables simultaneously—allowing for a much deeper, more complex understanding of a given scenario than you'll get with bivariate analysis.



There is NO any strong correlation are observed.

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.

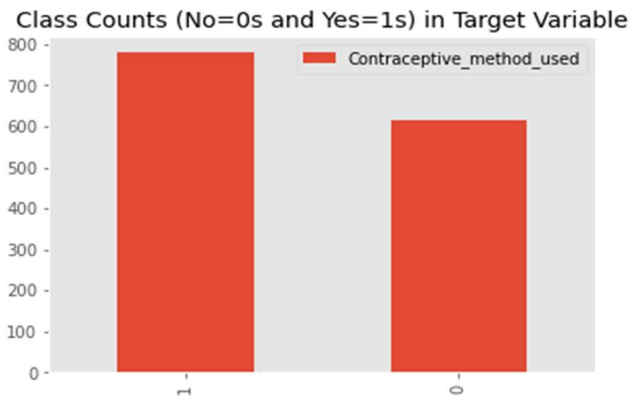
Encoding of Data:

Wife_age	fe_ educat	band_educa	f_children	and_Occup	rd_of_livin	ptive_meth	ligion_Scie	e_working	xposure_Not-Exposed	
0	24	Primary	Secondary	3	2	High	0	1	0	0
1	45	Uneducate	Secondary	10	3	Very High	0	1	0	0
2	43	Primary	Secondary	7	3	Very High	0	1	0	0
3	42	Secondary	Primary	9	3	High	0	1	0	0
4	36	Secondary	Secondary	8	3	Low	0	1	0	0

Further Encoding of Ordinal data as per guidelines:

Wife_age	fe_ educat	band_educa	f_children	and_Occup	rd_of_livin	ptive_meth	ligion_Scie	e_working	xposure_Not-Exposed	
0	24	2	3	3	2	3	0	1	0	0
1	45	1	3	10	3	4	0	1	0	0
2	43	2	3	7	3	4	0	1	0	0
3	42	3	2	9	3	3	0	1	0	0
4	36	3	3	8	3	2	0	1	0	0

Analyzing Target variable Contraceptive_method_used



Extracting the target column into separate vectors for training set and test set

Checking with top 5 data of dataset separated

Wife_age	fe_ educat	and_educ	f_children	and_Occup	rd_of_livin	ligion_Scie	e_working	posure_Not-Exposed	
0	24	2	3	3	2	3	1	0	0
1	45	1	3	10	3	4	1	0	0
2	43	2	3	7	3	4	1	0	0
3	42	3	2	9	3	3	1	0	0
4	36	3	3	8	3	2	1	0	0

2.Making the data split into train and test 70:30 ratio as per instruction

The shape of data of training and test

```
X_train (975, 9)
X_test (418, 9)
y_train (975,)
y_test (418,)
Total Obs 1393
```

Performing Logistic Regression:

```
#Formulation of LOGISTIC REGRESSION model on the train data.
model=LogisticRegression(max_iter=1000)
model.fit(X_train, y_train)
 y_predict=model.predict(X_test)
```

```
Accuracy Score on Train data is 0.6475336322869956
Accuracy Score on Test data is  0.6363636363636364
Accuracy of logistic regression classifier on train set: 0.65
Accuracy of logistic regression classifier on test set:  0.64
```

The classification Report and Confusion matrix:

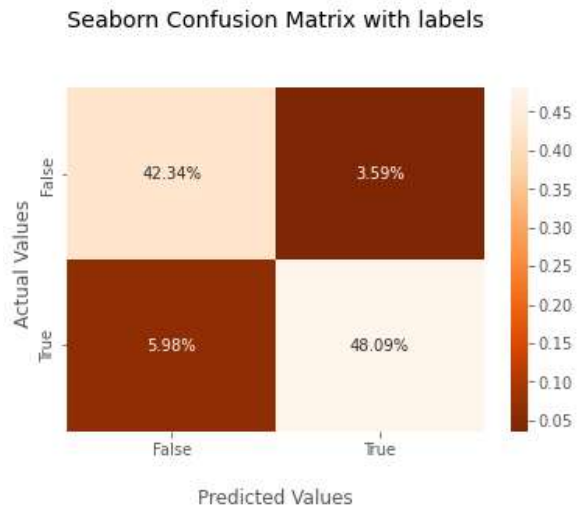
```
Classification Report
              precision    recall  f1-score   support

           0       0.88      0.92      0.90       192
           1       0.93      0.89      0.91       226

    accuracy                           0.90       418
   macro avg       0.90      0.91      0.90       418
weighted avg       0.91      0.90      0.90       418
```

Confusion Matrix

```
[[177 15]
[ 25 201]]
```



Tuning of Model by running K Fold

```
 [0.638755980861244, 0.6291866028708134, 0.638755980861244, 0.6267942583732058, 0.6363636363636364]
```

Mean of testing accuracy over 5 folds = 0.63 with std = 0.01

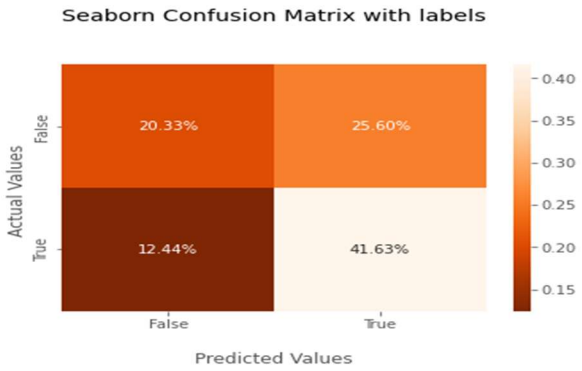
LINEAR DISCRIMINANT MODEL/ANALYSIS(LDA)

The classification report and Confusion matrix on test data

	precision	recall	f1-score	support
0	0.65	0.47	0.55	192
1	0.64	0.78	0.70	226
accuracy			0.64	418
macro avg	0.64	0.63	0.63	418
weighted avg	0.64	0.64	0.63	418

Confusion Matrix

```
[[ 91 101]
 [ 49 177]]
```



Building a Decision Tree Classifier (CART)

#Variable Importance

```
Wife_age                    0.31
No_of_children_born         0.23
Standard_of_living_index    0.10
Wife_ education             0.08
Husband_Occupation          0.08
Husband_education           0.08
Wife_working_Yes            0.05
Wife_religion_Scientology   0.04
Media_exposure_Not-Exposed  0.03
```

Regularizing the Decision Tree

```
ytrain_predict (1115,)
ytest_predict (418,)
y predict prob on test data.
array([[0.37019231, 0.62980769],
       [0.37019231, 0.62980769],
       [0.37019231, 0.62980769],
       [0.37019231, 0.62980769],
       [0.37019231, 0.62980769],
       [0.66784452, 0.33215548],
       [0.66784452, 0.33215548],
       [0.37019231, 0.62980769],
       [0.66784452, 0.33215548],
       [0.66784452, 0.33215548],
       [0.66784452, 0.33215548],
       [0.37019231, 0.62980769],
       [0.37019231, 0.62980769],
       [0.66784452, 0.33215548],
       [0.37019231, 0.62980769],
       [0.37019231, 0.62980769],
       [0.66784452, 0.33215548],
       [0.37019231, 0.62980769],
       [0.37019231, 0.62980769],
       [0.66784452, 0.33215548],
       [0.37019231, 0.62980769],
       [0.66784452, 0.33215548],
       [0.37019231, 0.62980769],
       [0.37019231, 0.62980769],
       [0.37019231, 0.62980769],
       [0.66784452, 0.33215548],
       [0.37019231, 0.62980769],
       [0.37019231, 0.62980769],
       [0.37019231, 0.62980769],
       [0.37019231, 0.62980769],
       [0.66784452, 0.33215548],
```

[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.66784452, 0.33215548],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.66784452, 0.33215548],
[0.66784452, 0.33215548],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.66784452, 0.33215548],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.66784452, 0.33215548],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.66784452, 0.33215548],
[0.66784452, 0.33215548],
[0.66784452, 0.33215548],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.66784452, 0.33215548],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.66784452, 0.33215548],
[0.37019231, 0.62980769],
[0.66784452, 0.33215548],
[0.37019231, 0.62980769],
[0.66784452, 0.33215548],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.66784452, 0.33215548],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.66784452, 0.33215548],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.66784452, 0.33215548],

[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.66784452, 0.33215548],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.66784452, 0.33215548],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.66784452, 0.33215548],
[0.37019231, 0.62980769],
[0.66784452, 0.33215548],
[0.37019231, 0.62980769],
[0.66784452, 0.33215548],
[0.66784452, 0.33215548],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.66784452, 0.33215548],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.66784452, 0.33215548],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.66784452, 0.33215548],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.66784452, 0.33215548],
[0.66784452, 0.33215548],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.66784452, 0.33215548],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.66784452, 0.33215548],
[0.66784452, 0.33215548],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],

```
[0.66784452, 0.33215548],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.66784452, 0.33215548],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.66784452, 0.33215548],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.66784452, 0.33215548],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.66784452, 0.33215548],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.66784452, 0.33215548],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.66784452, 0.33215548],
[0.66784452, 0.33215548],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.66784452, 0.33215548],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.66784452, 0.33215548],
[0.66784452, 0.33215548],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.66784452, 0.33215548],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.66784452, 0.33215548],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.66784452, 0.33215548],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.66784452, 0.33215548],
[0.37019231, 0.62980769],
```

```
[0.37019231, 0.62980769],
[0.66784452, 0.33215548],
[0.66784452, 0.33215548],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.66784452, 0.33215548],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.66784452, 0.33215548],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.66784452, 0.33215548],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.66784452, 0.33215548],
[0.37019231, 0.62980769],
[0.66784452, 0.33215548],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.66784452, 0.33215548],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.66784452, 0.33215548],
[0.66784452, 0.33215548],
[0.66784452, 0.33215548],
[0.66784452, 0.33215548],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.66784452, 0.33215548],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
```

[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.66784452, 0.33215548],
[0.66784452, 0.33215548],
[0.66784452, 0.33215548],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.66784452, 0.33215548],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.66784452, 0.33215548],
[0.66784452, 0.33215548],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.66784452, 0.33215548],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.66784452, 0.33215548],
[0.66784452, 0.33215548],
[0.37019231, 0.62980769],
[0.66784452, 0.33215548],
[0.37019231, 0.62980769],
[0.66784452, 0.33215548],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.66784452, 0.33215548],
[0.66784452, 0.33215548],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.66784452, 0.33215548],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.66784452, 0.33215548],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.66784452, 0.33215548],
[0.66784452, 0.33215548],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.66784452, 0.33215548],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.66784452, 0.33215548],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],

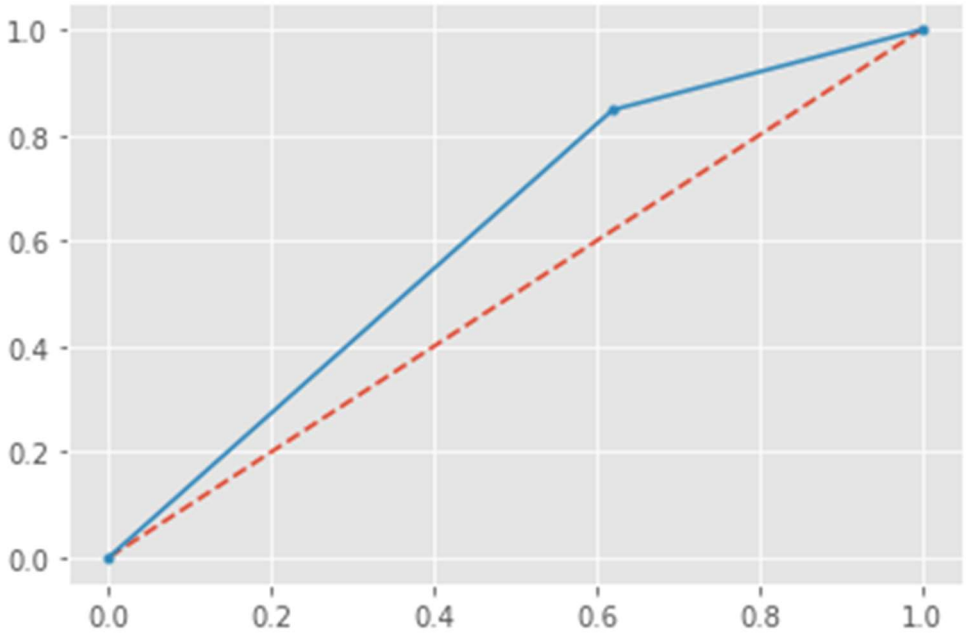
```
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.66784452, 0.33215548],
[0.66784452, 0.33215548],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.66784452, 0.33215548],
[0.37019231, 0.62980769],
[0.66784452, 0.33215548],
[0.66784452, 0.33215548],
[0.66784452, 0.33215548],
[0.66784452, 0.33215548],
[0.66784452, 0.33215548],
[0.37019231, 0.62980769],
[0.66784452, 0.33215548],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.66784452, 0.33215548],
[0.66784452, 0.33215548],
[0.66784452, 0.33215548],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.66784452, 0.33215548],
[0.66784452, 0.33215548],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.66784452, 0.33215548],
[0.37019231, 0.62980769],
[0.66784452, 0.33215548],
[0.37019231, 0.62980769],
```

```
[0.66784452, 0.33215548],
[0.37019231, 0.62980769],
[0.66784452, 0.33215548],
[0.37019231, 0.62980769],
[0.37019231, 0.62980769],
[0.66784452, 0.33215548],
[0.66784452, 0.33215548],
[0.66784452, 0.33215548],
[0.37019231, 0.62980769]])
```

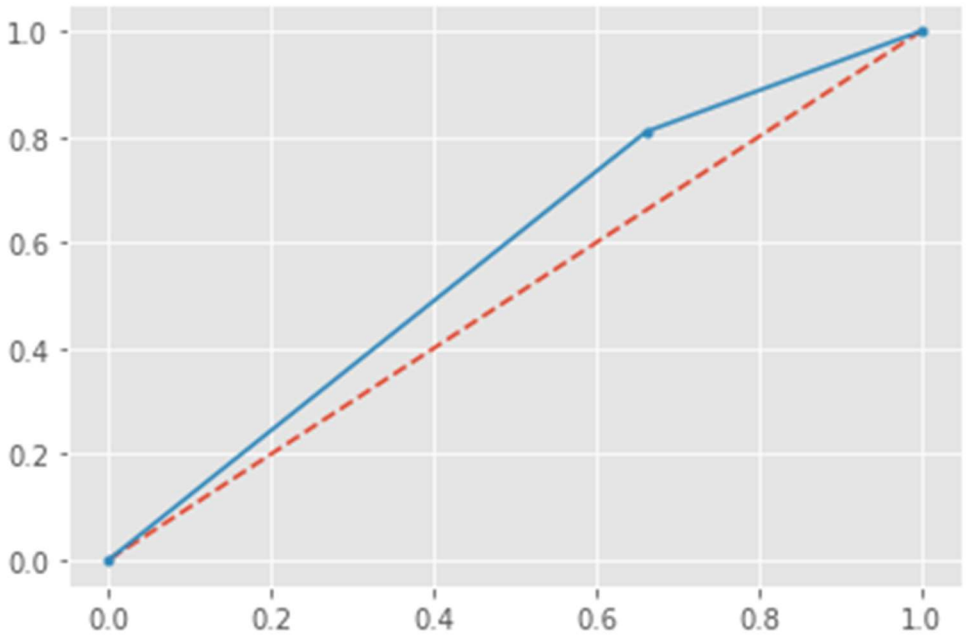
2.3 Performance Metrics: Check the performance of Predictions on Train and Test sets using Accuracy, Confusion Matrix, Plot ROC curve and get ROC_AUC score for each model Final Model: Compare Both the models and write inference which model is best/optimized.

Model Evaluation

AUC-ROC curve Analysis 0.614 on Train data



AUC-ROC curve Analysis 0.574 on Train data



Confusion Matrix for the training data

```
[[189, 308],
[ 94, 524]],
```

Training data Accuracy: **0.6394618834080718**

Classification report on training data

	precision	recall	f1-score	support
0	0.67	0.38	0.48	497
1	0.63	0.85	0.72	618
accuracy			0.64	1115
macro avg	0.65	0.61	0.60	1115
weighted avg	0.65	0.64	0.62	1115

Getting accuracy and summary of on different cuts in CART

Classification Report of the training data for cutoff of 0.1 :

	precision	recall	f1-score	support
0	1.00	0.02	0.04	497
1	0.56	1.00	0.72	618
accuracy			0.56	1115
macro avg	0.78	0.51	0.38	1115
weighted avg	0.76	0.56	0.42	1115

Classification Report of the training data for cutoff of 0.2 :

	precision	recall	f1-score	support
0	0.93	0.07	0.14	497
1	0.57	1.00	0.73	618
accuracy			0.58	1115
macro avg	0.75	0.53	0.43	1115
weighted avg	0.73	0.58	0.46	1115

Classification Report of the training data for cutoff of 0.3 :

	precision	recall	f1-score	support
0	0.88	0.21	0.34	497
1	0.61	0.98	0.75	618
accuracy			0.63	1115
macro avg	0.74	0.59	0.54	1115
weighted avg	0.73	0.63	0.57	1115

Classification Report of the training data for cutoff of 0.4 :

	precision	recall	f1-score	support
0	0.77	0.33	0.47	497
1	0.63	0.92	0.75	618
accuracy			0.66	1115
macro avg	0.70	0.63	0.61	1115
weighted avg	0.69	0.66	0.62	1115

Best Cutoff

Classification Report of the training data for cutoff of 0.45 :

	precision	recall	f1-score	support
0	0.71	0.42	0.53	497
1	0.65	0.86	0.74	618
accuracy			0.66	1115
macro avg	0.68	0.64	0.63	1115
weighted avg	0.67	0.66	0.65	1115

Confusion Matrix for the test data

[[65, 127],
[43, 183]]

Training data Accuracy **0.5933014354066986**

Classification report on test data

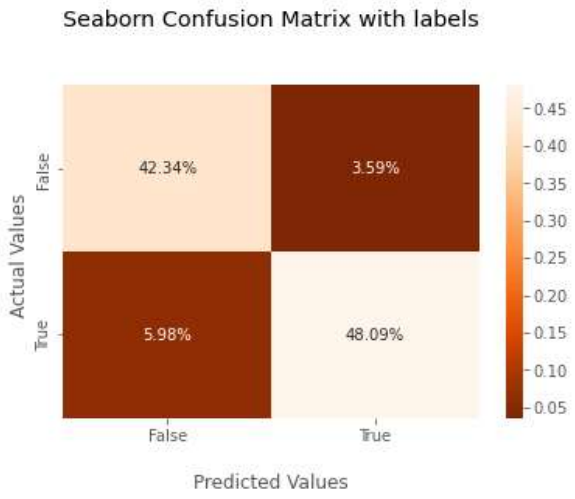
	precision	recall	f1-score	support

0	**0.60**	**0.34**	**0.43**	192
1	0.59	0.81	0.68	226
accuracy			**0.59**	418
macro avg	0.60	0.57	0.56	418
weighted avg	0.60	0.59	0.57	418

Model Comparison & Summary

Model Comparision					
Model		**Precision**	**Recall**	**f1-score**	**Accuracy**
Logistic R	0	0.88	0.92	0.43	**0.64**
	1	0.93	0.89	0.91	
LDA	0	0.65	0.47	0.55	**0.64**
	1	0.64	0.78	0.7	
CART	0	0.61	0.34	0.43	**0.59**
	1	0.59	0.81	0.68	

All there models major components are in above table for analysis. If we consider only accuracy than Logistic and LDA are better but ,if we analyze in light of Precision and Recall along with f1 Score than the model LDA will stand better to this case. There most of parameter are balance and well segregated even by its confusion matrix. Where parameters have been clearly distinguished. Data torture outcome is 48% of female using contraceptive while 3.56% are might using the contraceptive where as 42% are not using and almost 6% is prediction that they are not using but actually they are using.



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

Explain and summarize the various steps performed in this project.

For regression analysis we have followed the set rules and stages like:

Linear Regression: *Linear Regression represents some relationship between the dependent variable (y) and the independent variable (X).* It is used to classify the data.

We have implemented Logistic Regression in Python in just 7 steps:

Step 1: Imported the required libraries

Here, we have imported Pandas, NumPy, math, statsmodels, matplotlib and sklearn and other libraries to work with.

Step 2: Read the data using Pandas library

We have used pd.read_csv function to read the data.

There we did exploratory data analysis, the Data types, shape, Null Value detection EDA, 5-point summary. Performed Univariate, Bivariate Analysis, Multivariate Analysis, Outlier detection and treatment,

Then checked for categorical data and changed them to encoded mode for further analysis.

Step 3: Distribute the data into X and Y axis

We have distributed the columns of the data into X and y axis to visualize and predict the model.

Step 4: Split the data into train and test set

We have splited the data into the ratio of (7:3) as per guideline. That means, training data=70% and test data=30%.

Step 5: Fit the model and make prediction

We'll then fit the model using fit() method and will predict the data to build our model.

Step 6: Visualize the data using matplotlib

We'll visualize the data of the model using matplotlib library to have a better vision

Step 7: Calculated the accuracy of the model

By accuracy score, classification reports, precision, Recall analysis, Confusion Matrix analysis.

To Business

As business the focus should be on wife age and number of children born to find more that are using contraceptive or not.

Step by step for regressions:

Step 1	seaprating X and y
	X = df.drop('target', axis = 1)
	y = df['target']
Step 2	X_train, X_test, y_train, y_test = train_test_split(X, y, test_size =
Step 3	model = SomeRegressor()/Classifier()
Step 4	model.fit(X_train, y_train)
Step 5	y_pred_train = model.predict(X_train)
	y_pred_test= model.predict(X_test)
Step 6	classification_report(y_train, y_pred_train)
	classification_report(y_test, y_pred_test)
