Predictive Modelling project

By- Shubham Nagargoje

July22DSBA

Index:-

Problem 1……

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.

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.

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.

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

Problem 2……

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.

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.

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.

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

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.

Dataset for Problem 1: compactiv.xlsx

DATA DICTIONARY:

System measures used:

lread - Reads (transfers per second) between system memory and user memory
lwrite - writes (transfers per second) between system memory and user memory
scall - Number of system calls of all types per second
sread - Number of system read calls per second .
swrite - Number of system write calls per second .
fork - Number of system fork calls per second.
exec - Number of system exec calls per second.
rchar - Number of characters transferred per second by system read calls
wchar - Number of characters transfreed per second by system write calls
pgout - Number of page out requests per second
ppgout - Number of pages, paged out per second
pgfree - Number of pages per second placed on the free list.
pgscan - Number of pages checked if they can be freed per second
atch - Number of page attaches (satisfying a page fault by reclaiming a page in memory) per second
pgin - Number of page-in requests per second
ppgin - Number of pages paged in per second
pflt - Number of page faults caused by protection errors (copy-on-writes).
vflt - Number of page faults caused by address translation .
runqsz - Process run queue size (The number of kernel threads in memory that are waiting for a CPU to run.
Typically, this value should be less than 2. Consistently higher values mean that the system might be CPU-bound.)
freemem - Number of memory pages available to user processes
freeswap - Number of disk blocks available for page swapping.
----
usr - Portion of time (%) that cpus run in user mode

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

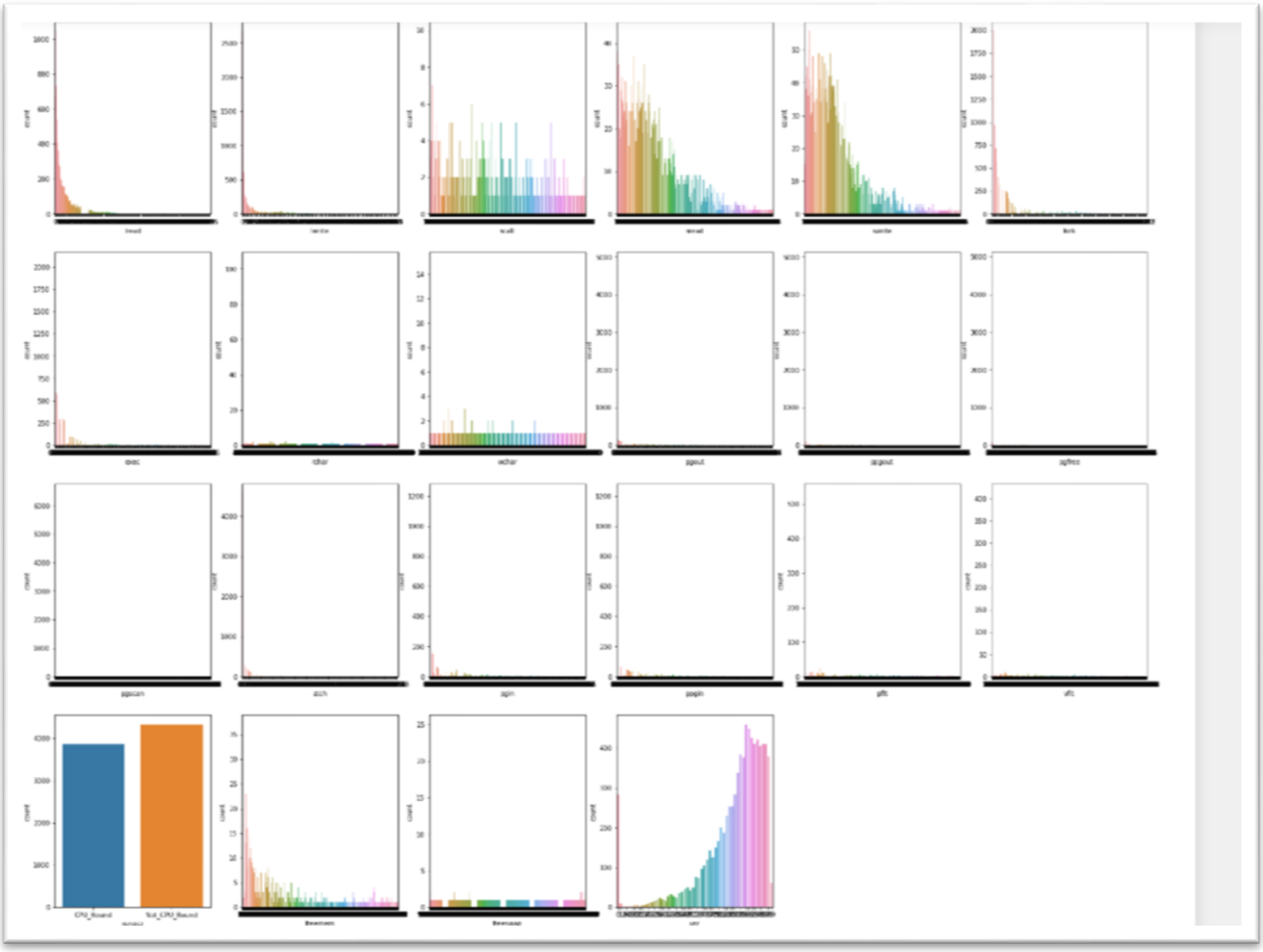
	lread	lwrite	scall	sread	swrite	fork	exec	rchar	wchar	pgout	...	pgscan	atch	pgin	ppgin	pflt	vflt	runqsz	freemem	freeswap
0	1	0	2147	79	68	0.2	0.2	40671.0	53995.0	0.0	...	0.0	0.0	1.6	2.6	16.00	26.40	CPU_Bound	4670	1730946
1	0	0	170	18	21	0.2	0.2	448.0	8385.0	0.0	...	0.0	0.0	0.0	0.0	15.63	16.83	Not_CPU_Bound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

5 rows × 22 columns

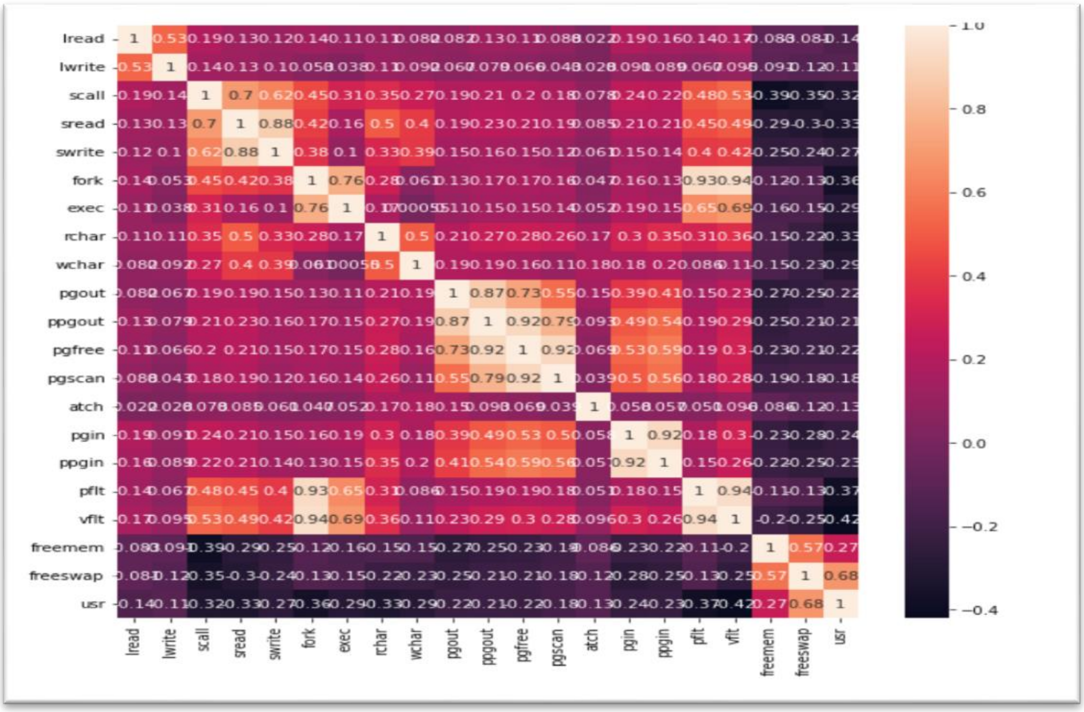
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8192 entries, 0 to 8191
Data columns (total 22 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   lread     8192 non-null   int64
 1   lwrite    8192 non-null   int64
 2   scall     8192 non-null   int64
 3   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
```

There are 21 numeric and 1 categorical variables in the dataset.

Univariate Analysis



Multivarite Analysis



There are some Null Values inside rchar and wchar variables and are treated with mean
There are 13 float , 8 int and 1 object
It has 8192 rows and 22 columns

Described dataset.

	count	mean	std	min	25%	50%	75%	max
lread	8192.0	1.342285e+01	15.159741	0.0	2.00	7.0	20.000	4.700000e+01
lwrite	8192.0	6.657471e+00	9.291945	0.0	0.00	1.0	10.000	2.500000e+01
scall	8192.0	2.294484e+03	1593.093446	109.0	1012.00	2051.5	3317.250	6.775125e+03
sread	8192.0	1.997764e+02	146.758932	6.0	86.00	166.0	279.000	5.685000e+02
swrite	8192.0	1.379700e+02	97.141835	7.0	63.00	117.0	185.000	3.680000e+02
fork	8192.0	1.557771e+00	1.591220	0.0	0.40	0.8	2.200	4.900000e+00
exec	8192.0	1.931495e+00	2.028253	0.0	0.20	1.2	2.800	6.700000e+00
rchar	8192.0	1.797970e+05	174495.517891	278.0	34860.50	127825.0	265394.750	6.111961e+05
wchar	8192.0	7.573578e+04	71257.347749	1498.0	22977.75	46653.0	106037.000	2.306259e+05
pgout	8192.0	1.420901e+00	2.200251	0.0	0.00	0.0	2.400	6.000000e+00
ppgout	8192.0	2.560702e+00	4.037317	0.0	0.00	0.0	4.200	1.050000e+01
pgfree	8192.0	3.164586e+00	4.983345	0.0	0.00	0.0	5.000	1.250000e+01
atch	8192.0	3.882788e-01	0.562937	0.0	0.00	0.0	0.600	1.500000e+00
pgin	8192.0	6.385262e+00	7.684420	0.0	0.60	2.8	9.765	2.351250e+01
ppgin	8192.0	9.140437e+00	11.160927	0.0	0.60	3.8	13.800	3.360000e+01
pflt	8192.0	1.056361e+02	101.548788	0.0	25.00	63.8	159.600	3.615000e+02
vflt	8192.0	1.756225e+02	162.497031	0.2	45.40	120.4	251.800	5.614000e+02
freemem	8192.0	1.387625e+03	1605.763418	55.0	231.00	579.0	2002.250	4.659125e+03
freeswap	8192.0	1.328520e+06	420782.723746	10989.5	1042623.50	1289289.5	1730379.500	2.243187e+06
usr	8192.0	8.624622e+01	9.748585	61.5	81.00	89.0	94.000	9.900000e+01

The five point summary of the data is given.

Mean mode median 25th percentile and 75th percentile values of all numeric variables are given.

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

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

```
for column in df.columns:
    if df[column].dtype != 'object':
        mean = df[column].mean()
        df[column] = df[column].fillna(mean)

df.isnull().sum()
```

```
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
```

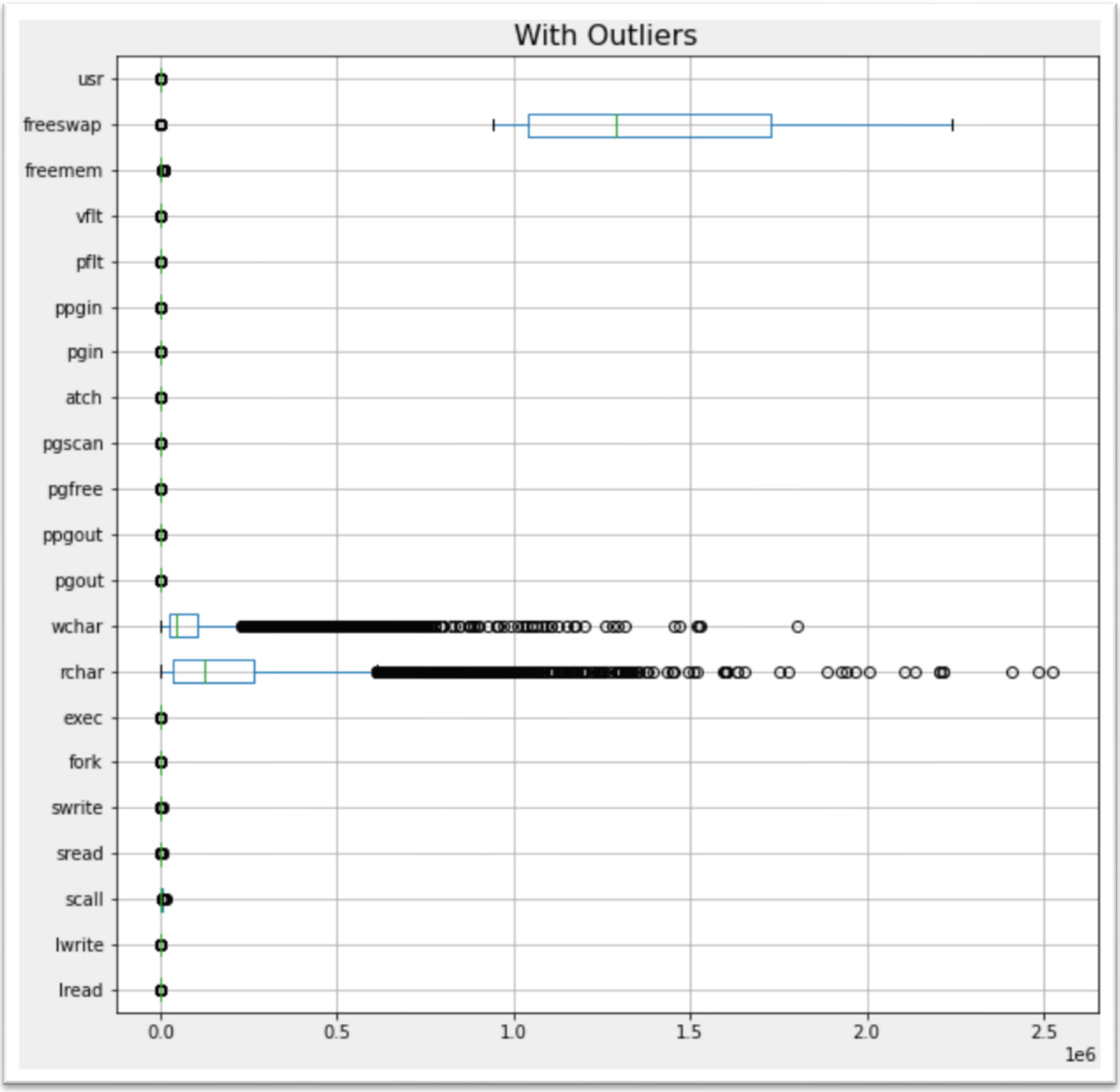
We checked for null values and duplicated data as they make a blunder with model building. So after finding null values , we imputed the values with the Median of the variable.

```
RUNQSZ :  2
CPU_Bound        3861
Not_CPU_Bound    4331
Name: runqsz, dtype: int64
```

For catergrical variable, RUNQSZ

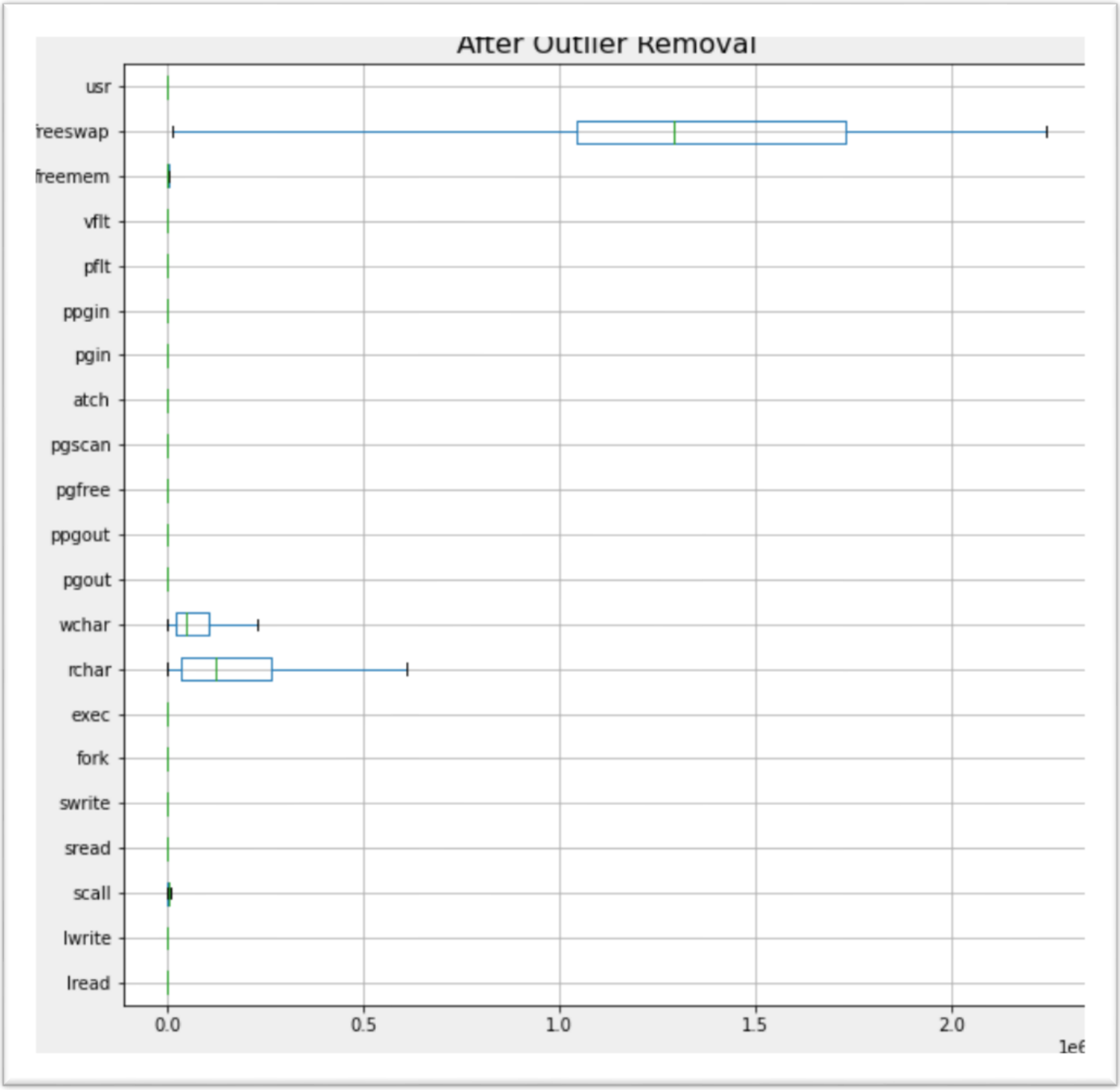
- We will encode the data and drop the first variable after encoding

<u>Checking Outliers.</u>



The following boxplot shows there are no. of outliers in the dataset present.

After Treating the outliers,



All the outliers are been treated successfully.

Lets check for duplicated data or rows.

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

There are no duplicated data present.

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.

```
# Copy all the predictor variables into X dataframe
X = df.drop('usr', axis=1)

# Copy target into the y dataframe.
y = df[['usr']]
```

Using USR as target variable , we define the X and Y variables.

```
# Split X and y into training and test set in 70:30 ratio
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.30 , random_state=1)
```

Data is plitted into testing and training data in 70:30 ratio

```
regression_model = LinearRegression()
regression_model.fit(X_train, y_train)

LinearRegression()
```

Here We fit the linear regression modelto the data set.

Following are the Coefficients with related variables.

```
'he coefficient for lread is -0.06348150618201623
'he coefficient for lwrite is 0.048161287091411424
'he coefficient for scall is -0.000663828011167507
'he coefficient for sread is 0.0003082521031421554
'he coefficient for swrite is -0.005421822297635938
'he coefficient for fork is 0.029312727248891713
'he coefficient for exec is -0.3211664838985831
'he coefficient for rchar is -5.166841759473579e-06
'he coefficient for wchar is -5.4028752354282645e-06
'he coefficient for pgout is -0.36881906387284397
'he coefficient for ppgout is -0.07659768212743255
'he coefficient for pgfree is 0.08448414470560629
'he coefficient for pgscan is -3.3306690738754696e-16
'he coefficient for atch is 0.62757415748176
'he coefficient for pgin is 0.019987907678685957
'he coefficient for ppgin is -0.06733383975700766
'he coefficient for pflt is -0.03360282937752637
'he coefficient for vflt is -0.005463668798515459
'he coefficient for freemem is -0.00045846718794665694
'he coefficient for freeswap is 8.831840263030009e-06
'he coefficient for runqsz_Not_CPU_Bound is 1.6152978488249081
```

R square on Testing and traing data.

```
# R square on testing data
regression_model.score(X_test, y_test)

0.7677318597936044
```

```
# R square on training data
regression_model.score(X_train, y_train)

0.796108610127457
```

79.6% of the variation in the usr is explained by the predictors in the model for train set

```python
#RMSE on Training data
predicted_train=regression_model.fit(X_train, y_train).predict(X_train)
np.sqrt(metrics.mean_squared_error(y_train,predicted_train))
```

4.419536092979902

```python
#RMSE on Testing data
predicted_test=regression_model.fit(X_train, y_train).predict(X_test)
np.sqrt(metrics.mean_squared_error(y_test,predicted_test))
```

4.6522957041927295

<u>RMSE</u> on traning data is 4.41 and on testing is 4.6 which is quite high.

```python
model.summary()
```

OLS Regression Results

Dep. Variable:	usr	**R-squared:**	0.796
Model:	OLS	**Adj. R-squared:**	0.795
Method:	Least Squares	**F-statistic:**	1115.
Date:	Sun, 04 Dec 2022	**Prob (F-statistic):**	0.00
Time:	22:20:30	**Log-Likelihood:**	-16657.
No. Observations:	5734	**AIC:**	3.336e+04
Df Residuals:	5713	**BIC:**	3.350e+04
Df Model:	20		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	84.1217	0.316	266.106	0.000	83.502	84.741
lread	-0.0635	0.009	-7.071	0.000	-0.081	-0.046
lwrite	0.0482	0.013	3.671	0.000	0.022	0.074
scall	-0.0007	6.28e-05	-10.566	0.000	-0.001	-0.001
sread	0.0003	0.001	0.305	0.760	-0.002	0.002
swrite	-0.0054	0.001	-3.777	0.000	-0.008	-0.003
fork	0.0293	0.132	0.222	0.824	-0.229	0.288
exec	-0.3212	0.052	-6.220	0.000	-0.422	-0.220
rchar	-5.167e-06	4.88e-07	-10.598	0.000	-6.12e-06	-4.21e-06
wchar	-5.403e-06	1.03e-06	-5.232	0.000	-7.43e-06	-3.38e-06
pgout	-0.3688	0.090	-4.098	0.000	-0.545	-0.192
ppgout	-0.0766	0.079	-0.973	0.330	-0.231	0.078
pgfree	0.0845	0.048	1.769	0.077	-0.009	0.178
pgscan	4.002e-14	1.62e-16	247.538	0.000	3.97e-14	4.03e-14
atch	0.6276	0.143	4.394	0.000	0.348	0.908
pgin	0.0200	0.028	0.703	0.482	-0.036	0.076
ppgin	-0.0673	0.020	-3.415	0.001	-0.106	-0.029
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

This is Model summary:-

R_square = 0.796

Adj R_square = 0.795

As the p-value of some variable exceeds 0.05 , that is they have no impact on the target variable. so we will try by dropping them.

Model 2

Linear Regression Model 2

```
#"ppgout","pgfree","pgin","fork""sread"
df1=df.drop(["fork","ppgout","pgfree","pgin","sread"], axis=1)
df1
```

Model summary

```
model1.summary()
```

OLS Regression Results

Dep. Variable:	usr	R-squared:	0.786
Model:	OLS	Adj. R-squared:	0.785
Method:	Least Squares	F-statistic:	1047.
Date:	Sun, 04 Dec 2022	Prob (F-statistic):	0.00
Time:	14:47:42	Log-Likelihood:	-16752.
No. Observations:	5734	AIC:	3.355e+04
Df Residuals:	5713	BIC:	3.369e+04
Df Model:	20		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	83.0584	0.312	265.968	0.000	82.446	83.671
lread	-0.0488	0.009	-5.481	0.000	-0.066	-0.031
lwrite	0.0379	0.013	2.888	0.004	0.012	0.064
scall	-0.0007	6.41e-05	-10.226	0.000	-0.001	-0.001
sread	0.0012	0.001	1.107	0.268	-0.001	0.003
swrite	-0.0063	0.001	-4.286	0.000	-0.009	-0.003
fork	-0.0612	0.133	-0.461	0.645	-0.321	0.199
exec	-0.3005	0.052	-5.781	0.000	-0.402	-0.199
rchar	-4.942e-06	4.93e-07	-10.030	0.000	-5.91e-06	-3.98e-06
wchar	-5.384e-06	1.06e-06	-5.095	0.000	-7.46e-06	-3.31e-06
pgout	-0.4643	0.091	-5.107	0.000	-0.642	-0.286
ppgout	0.0442	0.081	0.544	0.586	-0.115	0.203
pgfree	0.0236	0.050	0.476	0.634	-0.074	0.121
atch	0.7731	0.142	5.428	0.000	0.494	1.052
pgin	0.0268	0.028	0.941	0.347	-0.029	0.083
ppgin	-0.0747	0.020	-3.776	0.000	-0.113	-0.036
pflt	-0.0327	0.002	-16.656	0.000	-0.037	-0.029

R_square = 0.786

Adj R_square = 0.785

R-Square has been dropped significantly , so we will consider the model with highest R square value. i.e Model1

```
model.summary()
```

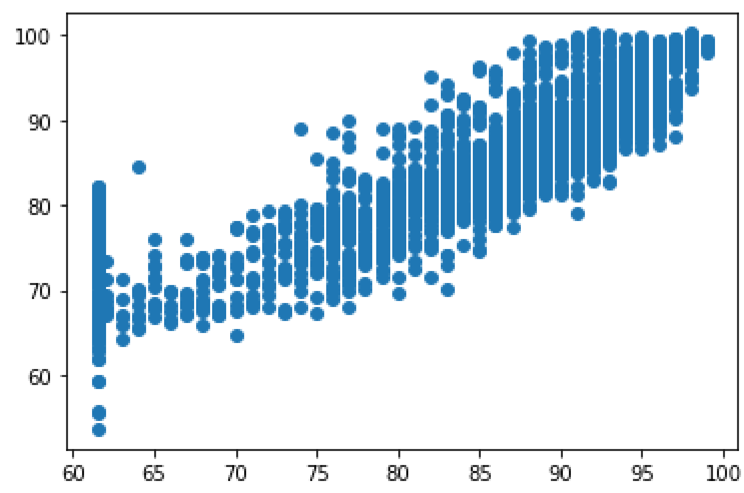
OLS Regression Results

Dep. Variable:	usr	**R-squared:**	0.796
Model:	OLS	**Adj. R-squared:**	0.795
Method:	Least Squares	**F-statistic:**	1116.
Date:	Sun, 04 Dec 2022	**Prob (F-statistic):**	0.00
Time:	14:47:50	**Log-Likelihood:**	-16656.
No. Observations:	5734	**AIC:**	3.335e+04
Df Residuals:	5713	**BIC:**	3.349e+04
Df Model:	20		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	84.1314	0.316	266.122	0.000	83.512	84.751
lread	-0.0634	0.009	-7.064	0.000	-0.081	-0.046
lwrite	0.0480	0.013	3.660	0.000	0.022	0.074
scall	-0.0007	6.28e-05	-10.576	0.000	-0.001	-0.001
sread	0.0003	0.001	0.336	0.737	-0.002	0.002
swrite	-0.0055	0.001	-3.805	0.000	-0.008	-0.003
fork	0.0296	0.132	0.225	0.822	-0.229	0.288
exec	-0.3211	0.052	-6.219	0.000	-0.422	-0.220
rchar	-5.212e-06	4.87e-07	-10.696	0.000	-6.17e-06	-4.26e-06
wchar	-5.346e-06	1.03e-06	-5.179	0.000	-7.37e-06	-3.32e-06
pgout	-0.3669	0.090	-4.077	0.000	-0.543	-0.190
ppgout	-0.0786	0.079	-0.999	0.318	-0.233	0.076
pgfree	0.0853	0.048	1.786	0.074	-0.008	0.179
atch	0.6304	0.143	4.414	0.000	0.350	0.910
pgin	0.0198	0.028	0.695	0.487	-0.036	0.076
ppgin	-0.0672	0.020	-3.406	0.001	-0.106	-0.029

Predictions plot

```
plt.scatter(y_test, y_pred)
plt.show()
```



The Final equation with coefficient is:-

Linear Equation ¶

```
for i,j in np.array(model.params.reset_index()):
    print('({}) * {} +'.format(round(j,3),i),end=' ')
```

```
(84.131) * const + (-0.063) * lread + (0.048) * lwrite + (-0.001) * scall + (0.0) * sread + (-0.005) * swrite + (0.03) * fork +
(-0.321) * exec + (-0.0) * rchar + (-0.0) * wchar + (-0.367) * pgout + (-0.079) * ppgout + (0.085) * pgfree + (0.63) * atch +
(0.02) * pgin + (-0.067) * ppgin + (-0.034) * pflt + (-0.005) * vflt + (-0.0) * freemem + (0.0) * freeswap + (1.614) * runqsz_N
ot_CPU_Bound +
```

usr= (83.06) * intercept + (-0.05) * lread + (0.04) * lwrite + (-0.0) * scall + (0.0) * sread + (-0.01) * swrite + (-0.06) * fork + (-0.3) * exec + (-0.0) * rchar + (-0.0) * wchar + (-0.46) * pgout + (0.04) * ppgout + (0.02) * pgfree + (0.77) * atch + (0.03) * pgin + (-0.07) * ppgin + (-0.03) * pflt + (-0.0) * vflt + (-0.0) * freemem + (0.0) * freeswap + (1.89) * runqsz_Not_CPU_Bound¶

There are some coefficient which are absolute 0 . including them in equation has no meaning so we will exclude them

The final equation after dropping is :-

usr= (83.06) * intercept + (-0.05) * lread + (0.04) * lwrite + (-0.01) * swrite + (-0.06) * fork + (-0.3) * exec + (-0.46) * pgout + (0.04) * ppgout + (0.02) * pgfree + (0.77) * atch + (0.03) * pgin + (-0.07) * ppgin + (-0.03) * pflt + (1.89) * runqsz_Not_CPU_Bound.

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

Following are the steps involved in the building the linear regression model:-
-Importing the dataset
-Checked for Null values and impurities in the data
-Outliers treatment to fit the model.
-converting all object type to integer
-splitting the data in train and test in 70:30 ratio
-Fitting the linear regression model.
-predicting the values and Accuracy of the model.
-use summary of the model to increase precession or recall
-re build the model by droping variables havi p-value>0.05
-re-check the model summary.
-consider the model with highest Accuracy as final model
-find coefficients and build the final equation

As per the final equation,

*usr= (83.06) * intercept + (-0.05) * lread + (0.04) * lwrite + (-0.01) * swrite + (-0.06) * fork + (-0.3) * exec + (-0.46) * pgout + (0.04) * ppgout + (0.02) * pgfree + (0.77) * atch + (0.03) * pgin + (-0.07) * ppgin + (-0.03) * pflt + (1.89) * runqsz_Not_CPU_Bound*

runqsz_Not_CPU_Bound has the highest Coefficient and swrite the least. so when runqsz increases by a unit value, usr tends to increase by 1.89

Problem 2: Logistic Regression, LDA and CART

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

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

Dataset for Problem 2: Contraceptive_method_dataset.xlsx

Data Dictionary:

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

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

EDA:-

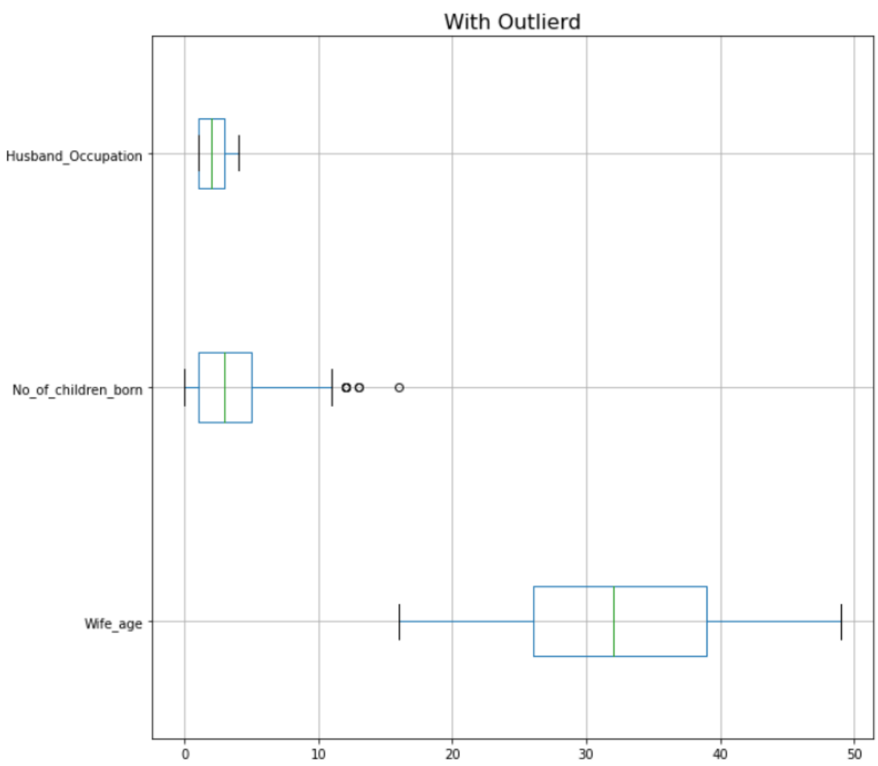
```
df.head()
```

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

```
<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
```

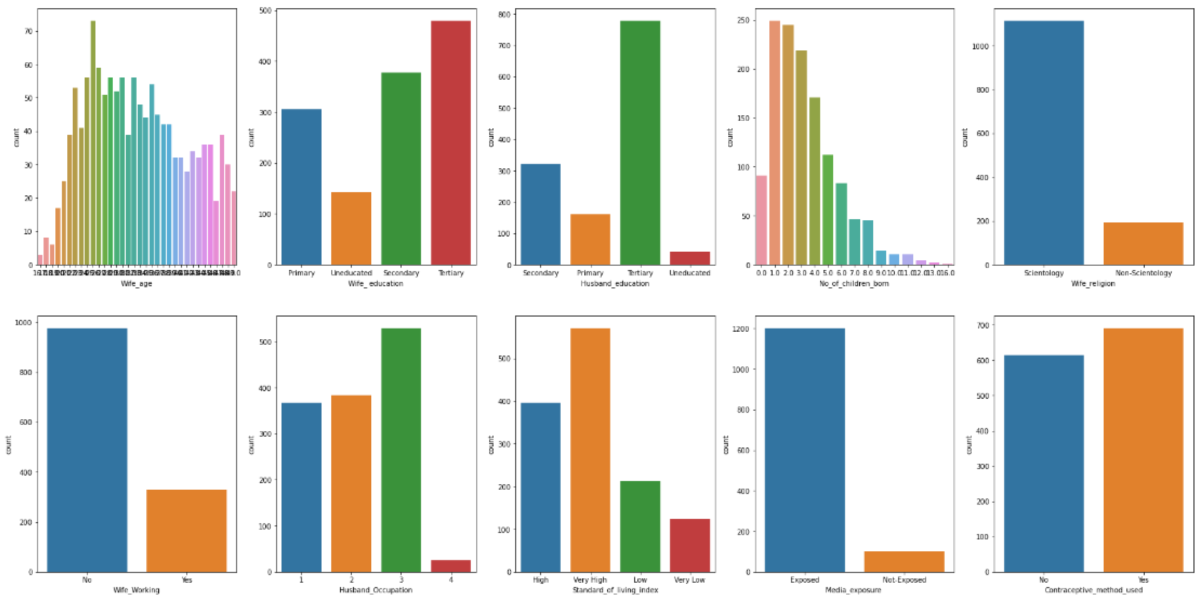
- There are 71 missing values in 'Wife_age' variable and 21 null values in ‘No_of_children_born’ variable
- There are 2 float, 1 integer and 7 object datatype present in the dataset.
- There are 1473 rows and 10 columns in the dataset.
- There are 80 duplicate rows found in the dataset.
- There are no anamolies in the categorical variable.

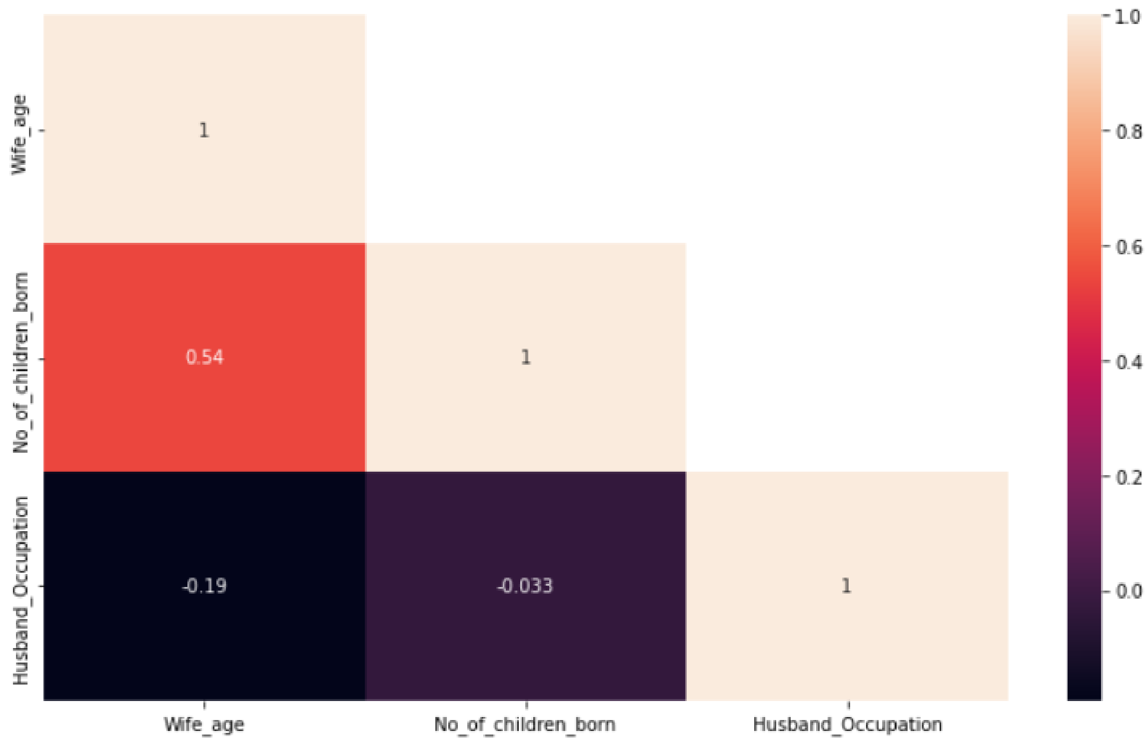
Let’s check if there are any outliers present or not.

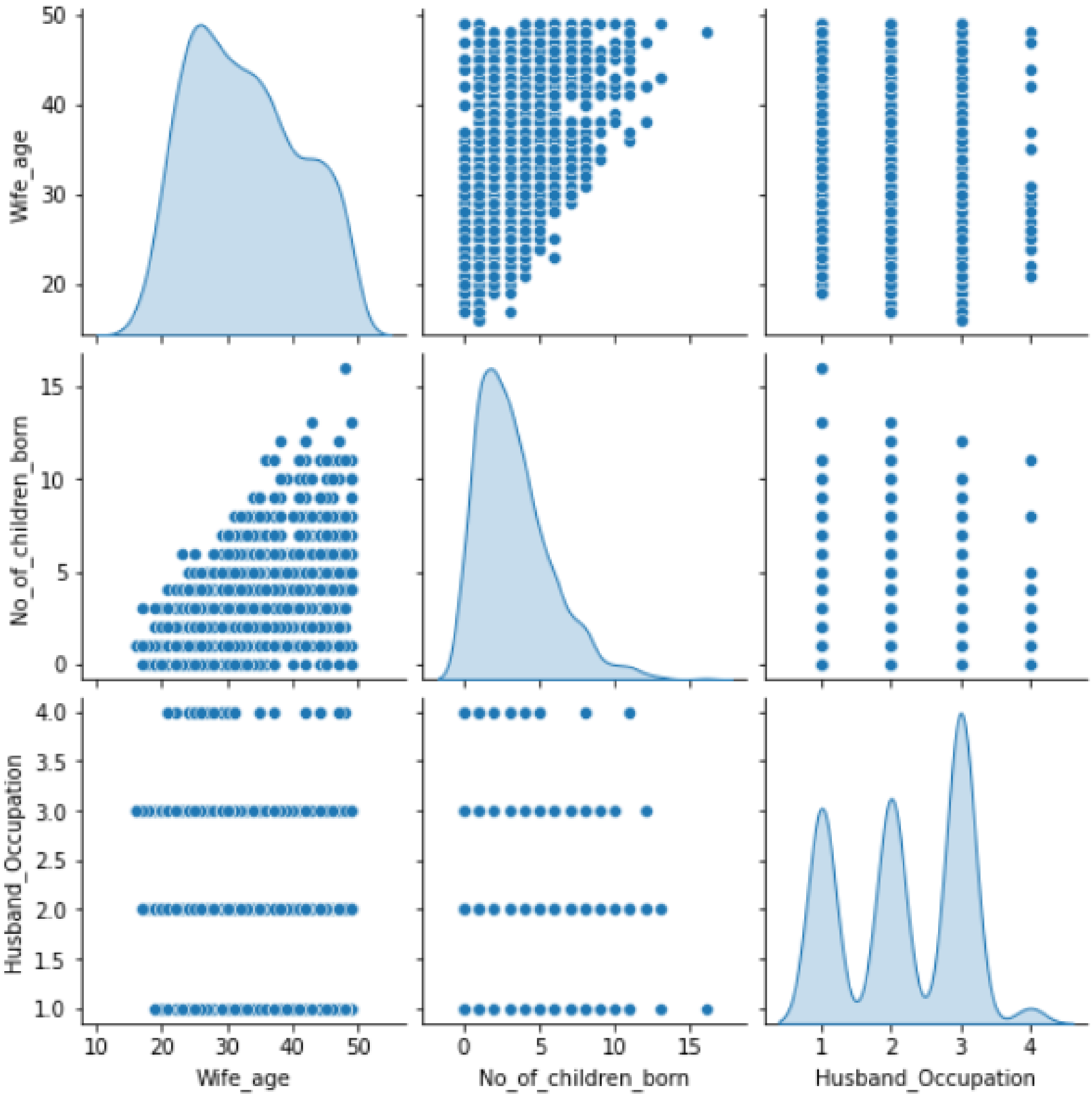


There is no need to treat outliers, as it won't affect the data.

Let's have a look at Univariate, bivariate and multivariate analysis.






Wife_age
No_of_children_born
Husband_Occupation
50
40
30
20
15
10
5
0
4.0
3.5
3.0
2.5
2.0
1.5
1.0
10
20
30
40
50
0
5
10
15
1
2
3
4
Wife_age
No_of_children_born
Husband_Occupation

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.

```
<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
```

- We have encoded the data i.e changed the datatype of the variables for further predictions.
- The variable "wife_education" was in object form , we converted it into integer datatype.
- The variable "Husband_education" was in categorical(object) form, it was converted to integer datatype.
- The variable "Wife_religion" was in object datatype, which was then converted to Integer datatype.
- The variable "Contraceptive_method_used", "wife_working", "Standard_of_living_index", "Media_Exposure " were in object datatype, and were converted to object datatype

```
## We are coding up the 'Wife_ education' variable in an ordinal manner

df_2['Wife_ education']=np.where(df_2['Wife_ education'] =='Uneducated', '1',df_2['Wife_ education'])
df_2['Wife_ education']=np.where(df_2['Wife_ education'] =='Primary', '2',df_2['Wife_ education'])
df_2['Wife_ education']=np.where(df_2['Wife_ education'] =='Secondary', '3',df_2['Wife_ education'])
df_2['Wife_ education']=np.where(df_2['Wife_ education'] =='Tertiary', '4',df_2['Wife_ education'])

#Husband_education
df_2['Husband_education']=np.where(df_2['Husband_education'] =='Uneducated', '1',df_2['Husband_education'])
df_2['Husband_education']=np.where(df_2['Husband_education'] =='Primary', '2',df_2['Husband_education'])
df_2['Husband_education']=np.where(df_2['Husband_education'] =='Secondary', '3',df_2['Husband_education'])
df_2['Husband_education']=np.where(df_2['Husband_education'] =='Tertiary', '4',df_2['Husband_education'])

#Wife_religion
df_2['Wife_religion']=np.where(df_2['Wife_religion'] =='Scientology', '1',df_2['Wife_religion'])
df_2['Wife_religion']=np.where(df_2['Wife_religion'] =='Non-Scientology', '0',df_2['Wife_religion'])

#Wife_Working
df_2['Wife_Working']=np.where(df_2['Wife_Working'] =='Yes', '1',df_2['Wife_Working'])
df_2['Wife_Working']=np.where(df_2['Wife_Working'] =='No', '0',df_2['Wife_Working'])

#Standard_of_living_index
df_2['Standard_of_living_index']=np.where(df_2['Standard_of_living_index'] =='Very Low', '1',df_2['Standard_of_living_index'])
df_2['Standard_of_living_index']=np.where(df_2['Standard_of_living_index'] =='Low', '2',df_2['Standard_of_living_index'])
df_2['Standard_of_living_index']=np.where(df_2['Standard_of_living_index'] =='High', '3',df_2['Standard_of_living_index'])
df_2['Standard_of_living_index']=np.where(df_2['Standard_of_living_index'] =='Very High', '4',df_2['Standard_of_living_index'])

#Media_exposure
df_2['Media_exposure ']=np.where(df_2['Media_exposure '] =='Exposed', '1',df_2['Media_exposure '])
df_2['Media_exposure ']=np.where(df_2['Media_exposure '] =='Not-Exposed', '0',df_2['Media_exposure '])

#Contraceptive_method_used
df_2['Contraceptive_method_used']=np.where(df_2['Contraceptive_method_used'] =='Yes', '1',df_2['Contraceptive_method_used'])
df_2['Contraceptive_method_used']=np.where(df_2['Contraceptive_method_used'] =='No', '0',df_2['Contraceptive_method_used'])
```

Train Test Split

```
# Copy all the predictor variables into X dataframe
X2 = df_2.drop('Contraceptive_method_used', axis=1)

# Copy target into the y dataframe.
y2 = df_2['Contraceptive_method_used']
```

```
# Split X and y into training and test set in 70:30 ratio
X2_train, X2_test, y2_train, y2_test = train_test_split(X2, y2, test_size=0.30 , random_state=1)
```

Splitting the data

Logistic Model Matrix:-

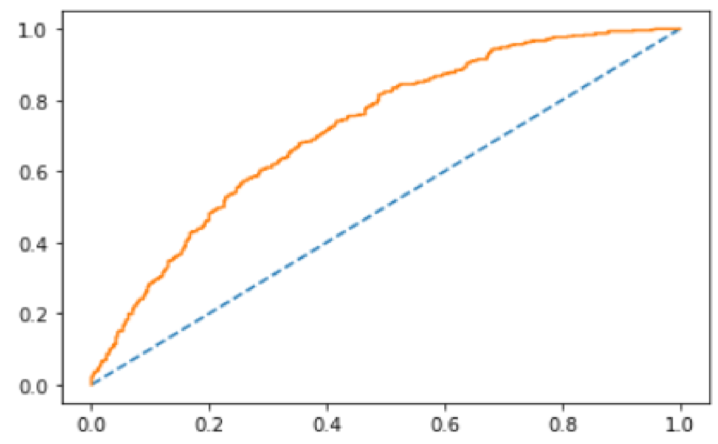
```
              precision    recall  f1-score   support

           0       0.72      0.49      0.58       205
           1       0.58      0.79      0.67       187

    accuracy                           0.63       392
   macro avg       0.65      0.64      0.63       392
weighted avg       0.66      0.63      0.62       392
```

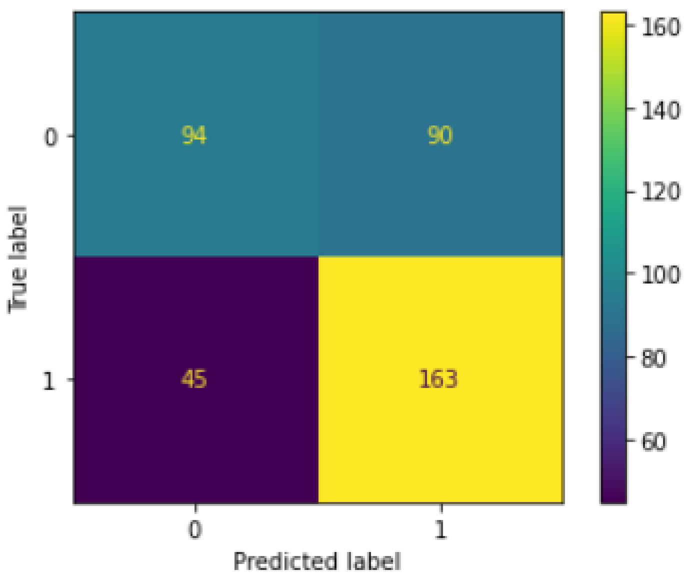
```
Classification Report of the training data:

              precision    recall  f1-score   support

           0       0.66      0.57      0.61       430
           1       0.66      0.75      0.70       483

    accuracy                           0.66       913
   macro avg       0.66      0.66      0.65       913
weighted avg       0.66      0.66      0.66       913
```

```
Classification Report of the test data:

              precision    recall  f1-score   support

           0       0.67      0.49      0.57       184
           1       0.64      0.79      0.71       208

    accuracy                           0.65       392
   macro avg       0.66      0.64      0.64       392
weighted avg       0.66      0.65      0.64       392
```

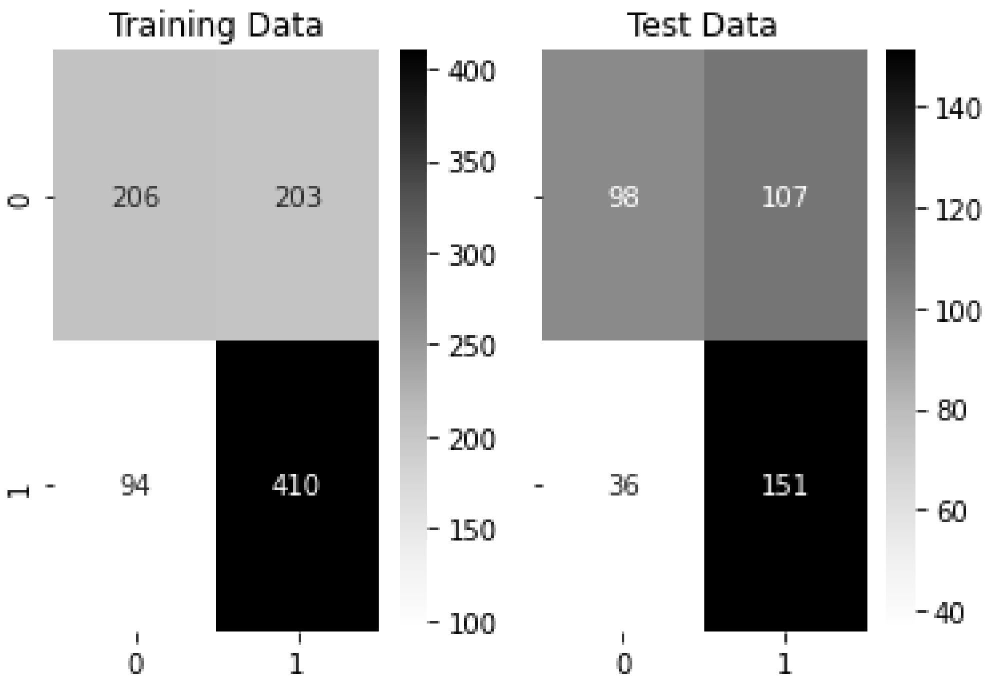
- Firstly, we have split the data into train and test.
- After that, Logistic Regression model was created, where we have fit the model and have predicted train and test for the dataset.
- Then we have checked the accuracy of train data which comes out to be 66% and of test data it was 65%
- After that, we have plot the AUC and ROC curve for the training data.
- Similarly, we have plotted the AUC and ROC Curve for the test data.

AUC: 0.720





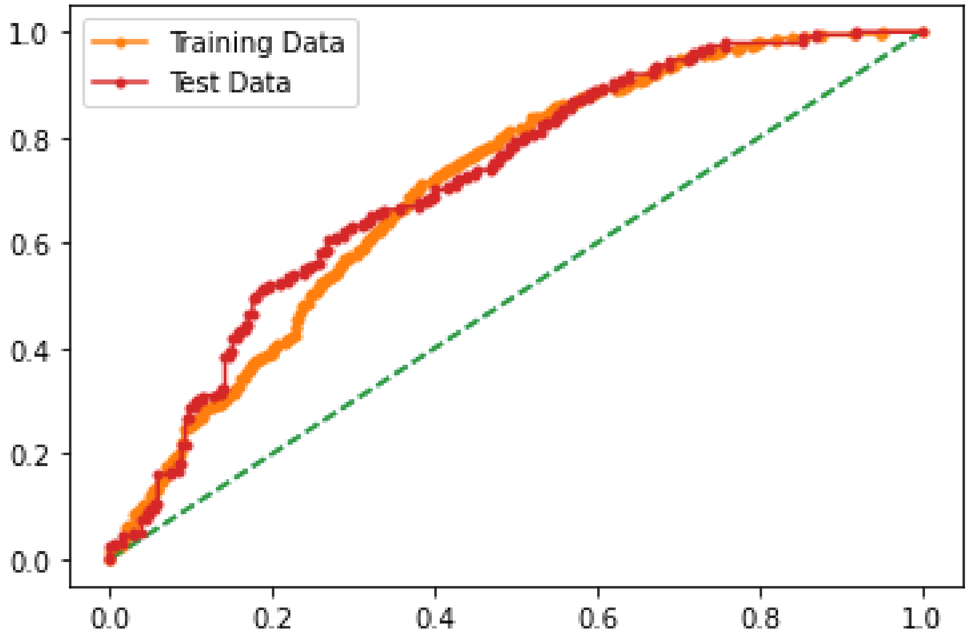
LDA MODEL:-




```
Classification Report of the training data:

              precision    recall  f1-score   support

           0       0.69      0.50      0.58       409
           1       0.67      0.81      0.73       504

    accuracy                           0.67       913
   macro avg       0.68      0.66      0.66       913
weighted avg       0.68      0.67      0.67       913

Classification Report of the test data:

              precision    recall  f1-score   support

           0       0.73      0.48      0.58       205
           1       0.59      0.81      0.68       187

    accuracy                           0.64       392
   macro avg       0.66      0.64      0.63       392
weighted avg       0.66      0.64      0.63       392
```

Probability prediction for the training and test data :

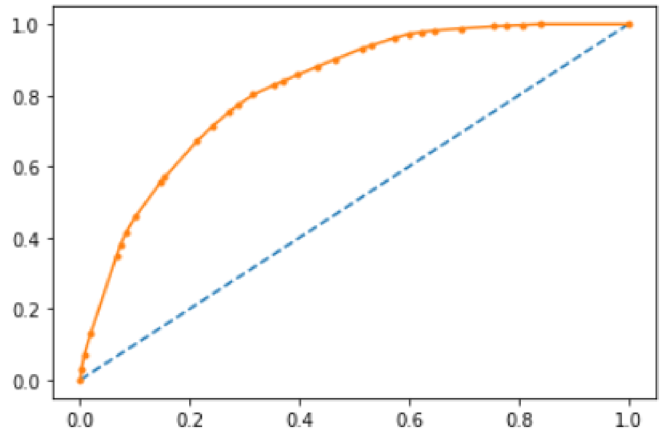
#AUC and ROC curve for training data as well as test data



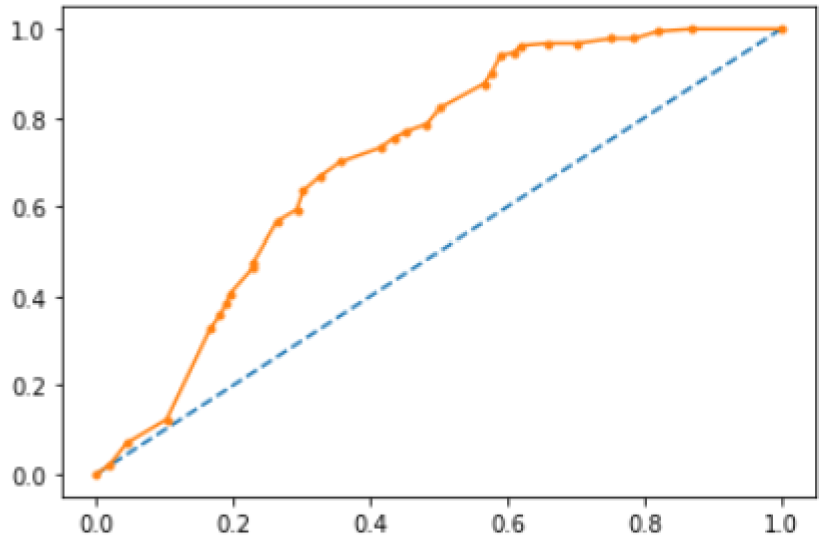
Decision Tree CART:-

AUC and ROC for the training data

AUC: 0.819



AUC: 0.712



Confusion Matrix for CART

```
print(classification_report(test_labels, ytest_predict))
```

```
              precision    recall  f1-score   support

           0       0.70      0.55      0.62       191
           1       0.64      0.78      0.70       201

    accuracy                           0.67       392
   macro avg       0.67      0.66      0.66       392
weighted avg       0.67      0.67      0.66       392
```

```
print(classification_report(train_labels, ytrain_predict))
```

```
              precision    recall  f1-score   support

           0       0.75      0.60      0.67       423
           1       0.71      0.83      0.76       490

    accuracy                           0.72       913
   macro avg       0.73      0.71      0.71       913
weighted avg       0.73      0.72      0.72       913
```

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.

Logistics Model:-

Inferences using the Contaceptive Methods used {No == 0}:

Precision (72%) – 72% of Customers who didnot used the contraceptive methods are correctly predicted ,out of all Customers who didnot used the contraceptive methods are predicted .

Recall (49%) – Out of all the Customers who actually didnot used the contraceptive methods , 95% of Customers who didnot Churn have been predicted correctly .

For { Contaceptive Methods used (yes == 1)}:

Precision (58%) – 48% of Customers who did used the contraceptive methods are correctly predicted ,out of all Customers who did used the contraceptive methods that are predicted .

Recall (79%) – Out of all the Customers who actually did used the contraceptive methods , 79% of Customers who did used the contraceptive methods have been predicted correctly .

Overall accuracy of the model – 63 % of total predictions are correct

For LDA Model:-

Inferences using the Contaceptive Methods used {No == 0}:

Precision (73%) – 73% of Customers who didnot used the contraceptive methods are correctly predicted ,out of all Customers who didnot used the contraceptive methods are predicted .

Recall (48%) – Out of all the Customers who actually didnot used the contraceptive methods , 48% of Customers who didnot Churn have been predicted correctly .

For { Contaceptive Methods used (yes == 1)}:

Precision (59%) – 59% of Customers who did used the contraceptive methods are correctly predicted ,out of all Customers who did used the contraceptive methods that are predicted .

Recall (81%) – Out of all the Customers who actually did used the contraceptive methods , 81% of Customers who did used the contraceptive methods have been predicted correctly .

Overall accuracy of the model – 64 % of total predictions are correct

Decision Tree (CART):-

Inferences using the Contaceptive Methods used {No == 0}:

Precision (70%) – 70% of Customers who didnot used the contraceptive methods are correctly predicted ,out of all Customers who didnot used the contraceptive methods are predicted .

Recall (55%) – Out of all the Customers who actually didnot used the contraceptive methods , 55% of Customers who didnot used the contraceptives have been predicted correctly .

For { Contaceptive Methods used (yes == 1)}:

Precision (64%) – 64% of Customers who did used the contraceptive methods are correctly predicted ,out of all Customers who did used the contraceptive methods that are predicted .

Recall (78%) – Out of all the Customers who actually did used the contraceptive methods , 78% of Customers who did used the contraceptive methods have been predicted correctly .

Overall accuracy of the model – 67 % of total predictions are correct

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

By comparing All the Models ,

DecisionTree Model is the best fit and optimized for the dataset.

```
In [ ]:
```