Date :- 11/04/22
10:30 pm

Multiple Linear Regression

#Multiple Linear Regression :-

Examines Relationship between [two] (or) [more Than two] Variables

* Each [independent Variable] has its own corresponding coefficient.

$$y = b_0 + b_1 * x_1 + b_2 * x_2 + \dots + b_n x_n$$

dependent Variable

Independent Variables.

⇒ Example :

TV	Radio	np	Sales

TV, Radio, news paper, Sales

$$\hat{y} = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta x_3$$

TV, News paper, newspaper

y- intercept :- $\hat{y}$

Coefficient :- $\beta_1, \beta_2, \beta_3$

Same Example : But different method

Step-1 Bussiness Problem Understanding.

* What is The relation b/w each adversting channel [TV, Radio, Newspaper] and Sales ? $y = f[x_1, x_2, x_3]$
* Previously, we Explored is There a relationship between Total adversting Spend and Sales ? as well as predicting the total Sales For Some Value of Total spend.

Step: 2.1 Data collection

```
# df = pd.read_csv("Advertising.csv")
# df.head()
```

Out

TV	Radio	Newspaper	Sales
230.1	37.8	69.2	22.1
44.5	39.3	45.1	10.4
17.2	45.9	69.3	9.3
151.5	41.3	58.5	18.5
180.8	10.8	58.4	12.9

Step 2.2 Data Understanding

Same as Simple linear Regression Dataset.

Step 2.3 Data Set Understanding

```
# df.info()
```

Step-3.1

Exploratory Data Analysis [EDA]

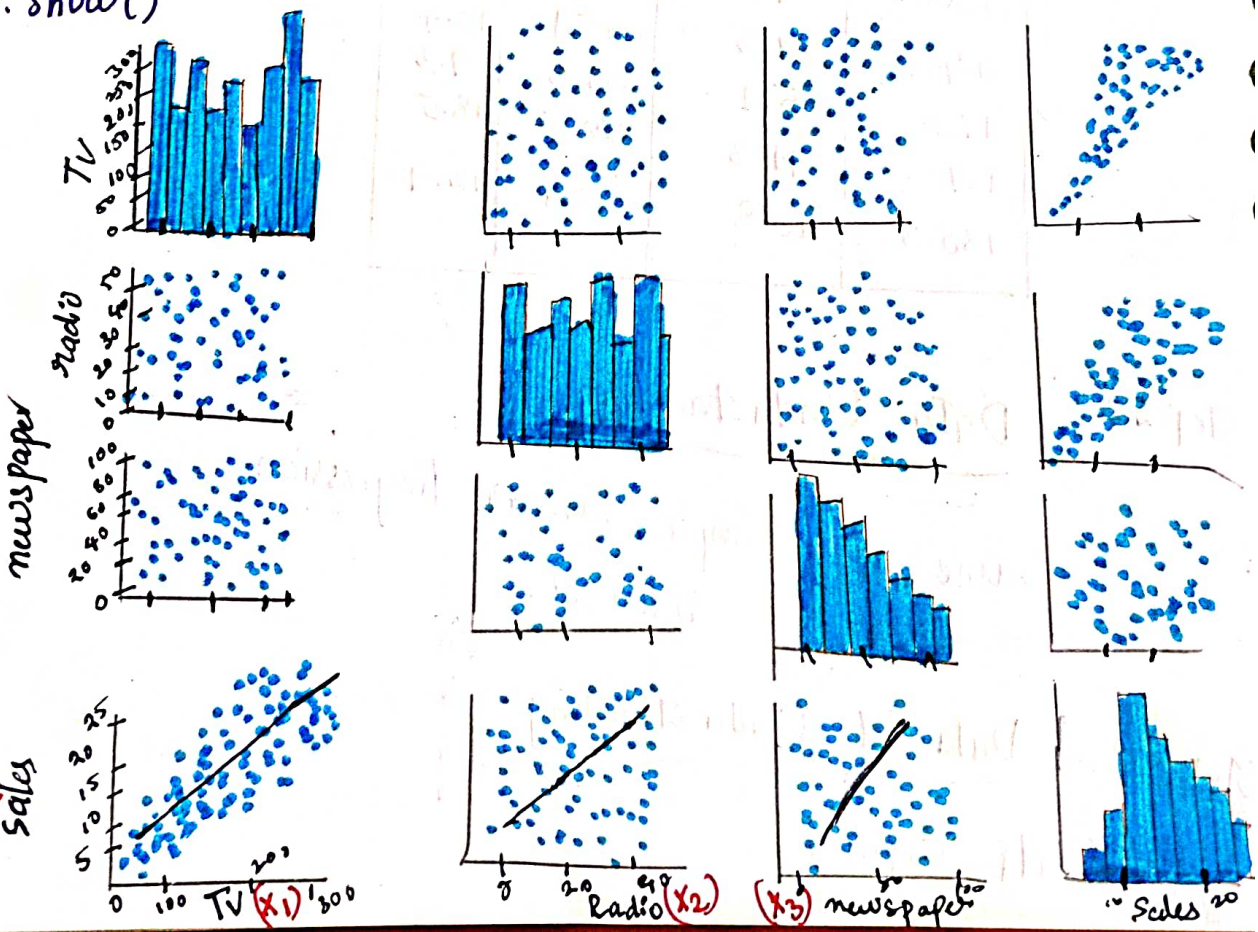
df.describe()

Out:

	TV	Radio	newspaper	Sales
count	200.000000	200.000000	200.000000	200.000000
Mean	147.042500	23.264000	30.554000	14.022500
Std	85.854236	14.846809	21.778621	5.217457
min	0.700000	0.000000	0.300000	1.600000
25%	74.375000	9.975000	12.750000	10.375000
50%	149.750000	22.900000	25.750000	12.900000
75%	218.825000	36.525000	45.100000	17.400000
max%	296.40000	49.600000	114.000000	27.000000

sns.pairplot(df)

plt.show()



⇒ By observing the scatter plot, we made an assumption of relation between [y] and [$x_1 + x_2 + x_3$] is Linear

(symmertical Matrix)

df.corr() →

	TV	Radio	News paper	Sales
TV	1.000000	0.054809	0.056648	0.782224
Radio	0.054809	1.000000	0.354104	0.576223
News paper	0.056648	0.354104	1.000000	0.228299
Sales	0.782224	0.576223	0.228299	1.000000

* Where upper Triangle, lower Triangle is same
The diagonal elements = 1

i) y (vs) x_1
ii) y [vs] x_2
iii) y [vs] x_3
→ relation strong → High Accuracy

x_1 [vs] x_2
x_1 (vs) x_3
x_2 [vs] x_3
→ Low ↓
If strong
collinearity problem

⇒ The relation between "y" and "x" be high ...
dependent
Independent variable.

⇒ The higher the value - stronger the correlation

⇒ The relation between any two independent Variables should be "Low"

**** ⇒ if the correlation between any two (2) independent Variables is strong, then it is called "collinearity problem".

Should be "weak"

Should be weak

* idenpendent variable [student 2]

Should be strong

* idenpendent Variable [students]
* dependent Variable [Teacher]

Example

12/4/22
3:30 pm

Dt: 12/4/2
1:00pm

STEP: 3.2 Data cleaning

Same as Simple Linear

STEP 3.3 Data Wrangling

not Same •

STEP 3.4 Train_Test split

```
# X = df.drop(columns = "sales")
# y = df["sales"]

from sklearn.model_selection import train_test_split

# X_train, Xtest, y_train, y_test = train_test_split(x, y,
                                    test_size = 0.3, Random_
                                    state = 29)
```

STEP-4 Modelling

Here, $y = \beta_0 + \beta_x \times [TV] + \beta_2 \times [radio] + \beta_3 \times [Newspaper]$

```
from sklearn.linear_model import LinearRegression
                                  ↑ capital
# model = LinearRegression()
# model.fit(x_train, y_train)
```

Out: LinearRegression()

model.coef_

Out : array([0.04422917, 0.18781641, 0.00758744])
β_1 β_2 β_2

model.intercept_

Out : 2.974097357
β_0

⇒ Predictions → Predicting on "X"

train_predictions = model.predict(x_train)

test_predictions = model.predict(x_test)

STEP: 5 Evaluations → on "y"

from sklearn.metrics import mean_squared_error

test_RMSE = np.sqrt(mean_squared_error(y_test, test_predictions))

train_RMSE = np.sqrt(mean_squared_error(y_train, train_predictions))

print(train_RMSE, test_RMSE)

Out : 1.64082 , 1.7805

model.score (Xtrain, y_train) (train R^2)

Out: 0.888817

model.score (X_test, y_test) (test R^2)

Out: 0.905258

⇒ checklist

Que: Is model has Underfitting (or) Overfitting problem?

Ans:- Good model

Que: Is Test Accuracy = Cross Validation Score

Ans:

```
from sklearn.model_selection import Cross_Val_score
# scores = cross_val_score (model, X, y, cv=5)
# print (scores)
# scores.mean()
```

Out [0.87865198, 0.9176312, 0.92933, 0.81443, 0.895...

→ 0.887106349, compare with Xtest, ytest Accuracy.

3. Check Assumptions

* Linearity of errors

test_res = y_test - test_predictions (Xtest)

↓ Residuals

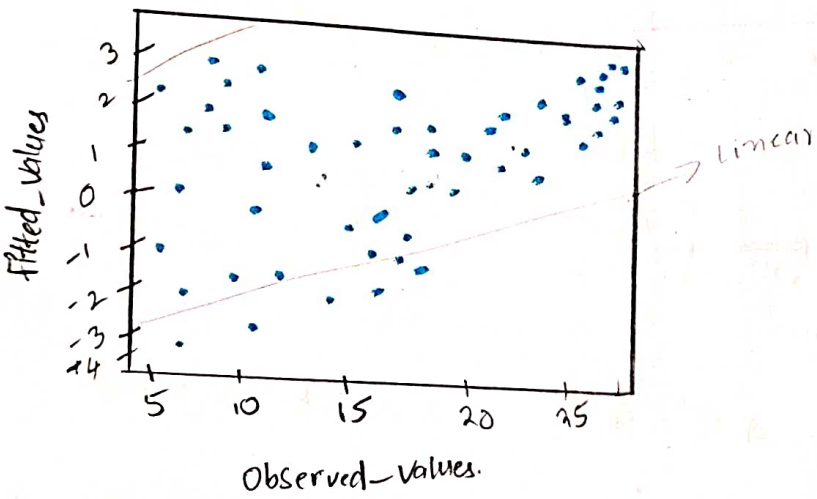
plt.scatter (y_test, test_res)

plt.xlabel ("Observed_values")

plt.ylabel ("fitted_values")

plt.show()

Out:



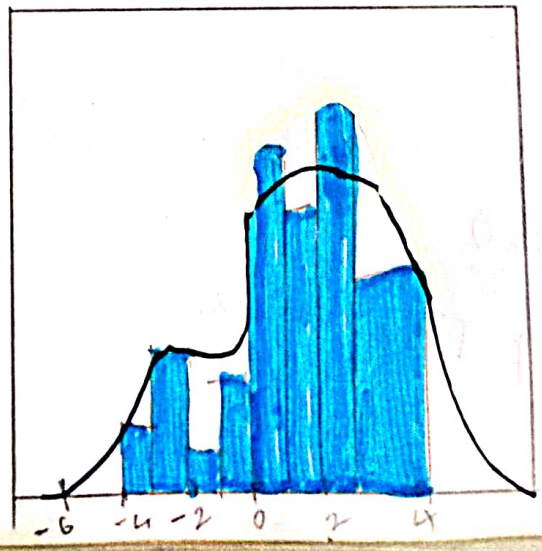
* Normality of errors

sns.distplot (test_res, bins = 15, kde = True)

plt.show()

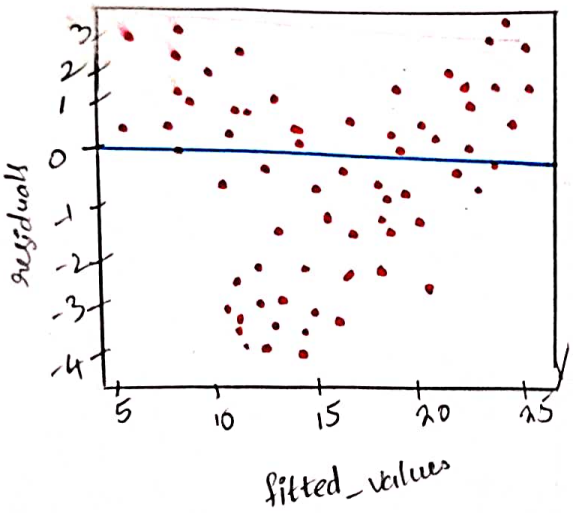
Out:

Left skewed →



3. Equal Variance of Errors (Homoscadesicity) ✗

```
# plt.scatter(test_predictions, test_res, c="r")
# plt.axhline(y=0, color="blue")
# plt.xlabel("fitted_values")
# plt.ylabel("residuals")
# plt.show()
```



* Assumptions Failed, $R^2 = 86\%$ [We reject the model]

⇒ Every condition in checklist should satisfy Than, Only we consider the model

4. Variable Significance *

$H_0 : \beta_1 = 0$

$H_1 : \beta_1 \neq 0$

⇒ Hypothesis Testing For Variables.

import Statsmodels.formula.api smf

```
# model 1 = smf.ols("y ~ x", data = df).fit()
# model 1.summary()
```

Out: OLS Regression Results.

Dep variable : Y
Model : OLS
method : Least Squars

R^2-squared : 0.897
Adj. R-squared : 0.896
F statistic : 570.3
Prob (F-statistic) : $1.58e^{-96}$

For Model: Accept H_0 :- 1.58×10^{-96} - $P < 0.05$
↳ Reject H_0

	Coeff	Std err	t	$P > \|t\|$	[0.025	0.975]
Intercept	2.9389	0.312	9.422	0.000	2.324	3.554
β_1 X(0)	$\neq 0$ 0.2458	0.001	32.809	0.000	0.043	0.049
β_2 X(1)	$\neq 0$ 0.1885	0.009	21.893	0.000	0.472	0.206
β_3 X(2)	$= 0$ -0.0010	0.006	-0.177	0.860	-0.013	0.011

Variable wise → $P < 0.05$ → P. low Null. go → Reject H_0

Variable 1
H_0 : $\beta_1 = 0$ ✗
H_1 : $\beta_1 \neq 0$ ✓

Variable 2
$\beta_2 = 0$ ✗
$\beta_2 \neq 0$ ✓

Variable 3
H_0 : $\beta_3 = 0$ ✓
H_1 $\beta_3 \neq 0$ ✗

$P > 0.05$

⇒ Checking whether data has any influence value using influence index plots

1) * Influence Index plots, For Making "p" value equal < 0.05)

```
import statsmodels.api as sm
# sm.graphics.influence_plot(model1)
```

Why?

$\beta_3 = 0$,

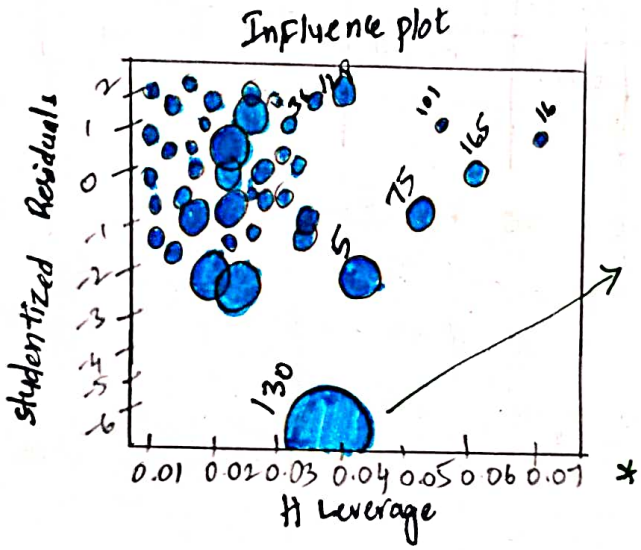
$\hat{y} = \beta_0 + \beta_1 x + \beta_2 x_2 + \beta_3 x_3$

By dropping, we are losing 1/3 of data = 33% data

In That Case, we drop some records, That are influencing More.

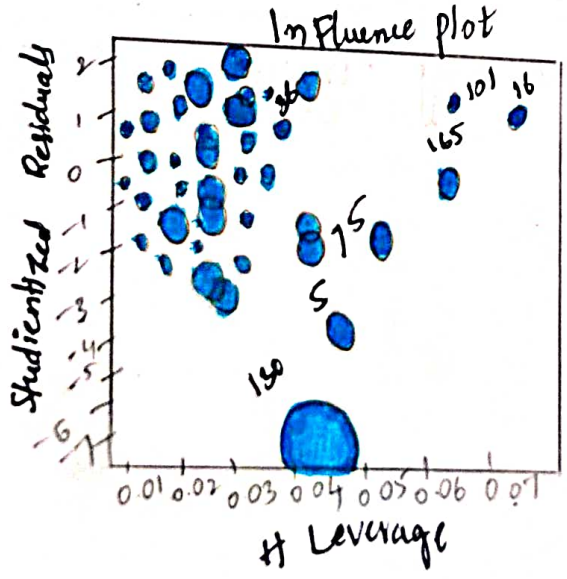
Ex:- 1/200 = 0.05%.

Out:



Index 130 is showing high influence, So, we exclude that entire row.

* Studentized Residuals = Residuals / standard deviation of residuals.



row

```
# df_new = df.drop(df.index[[130]], axis=0)
# df_new.
```

Out

	Tv	Radio	News Paper	Sales
0	230.1	37.8	69.2	22.1
1	44.5	39.3	45.1	10.4
⋮	⋮	⋮	⋮	⋮
199	232.1	8.6	8.7	13.4

199 x 4 columns

Once again Rebuild model

```
# model 2 = smf.ols(formula = "Sales ~ Tv + radio + newspaper",
                    data = df_new).fit()

# model2.summary()
```

Out : OLS Regression MODEL

→ This is one method of reducing the P value.

	Coeff	Std.err	t	P>|t|
Intercept	3.0931	0.290	10.654	0.000
Tv X(0)	0.0448	0.001	34.425	0.000
radio X(1)	0.1939	0.009	24.130	0.000
news paper X(2)	-0.043	0.005	-0.777	0.438

→ Reduced From 0.86 to 0.43 By Reducing.

2)

⇒ Variance inflation Factor [VIF]

Variance inflation Factor [VIF] measures ratio between the variance for a given regression coefficient with only that variable in the model versus the variance for given regression coefficient with all variables in the model.

1 independent variable influence on other independent variable is called as ("VIF")

affecting

X_1 X_2 X_3

TV → radio → news

TV ← Radio → news

TV ← Radio ← news

$$V.I.F = \frac{1}{1-R^2}$$

```
# rsq_Tv = smf.ols ("Tv ~ radio + newspaper", data = df)
                                              .fit()

# rsq_Tv.summary()
```

Out : OLS Regression Results

R-squared = 0.005

$$V.IF = \frac{1}{1-(0.05)^2}$$

V.IF = 1.0000

Calculating VIF's values of independent variables

```
# rsq_Tv = smf.ols ("Tv ~ radio + newspaper", data = df).
                                    .fit().rsquared

# Vif_Tv = 1/(1 - rsq_Tv)
```

```
# rsq_radio = smf.ols("radio ~ TV + newspaper", data=df)
                .fit().rsquared

# vif_radio = 1/(1-rsq_radio)

.....

# rsq_newspaper = smf.ols("newspaper ~ radio + TV",
                  data = df).fit().rsquared

# vif_newspaper = 1/(1-rsq_newspaper)

.....

Storing VIF values in a dataFrame

# d1 = {"variables": ["TV", "radio", "newspaper"],
        "VIF": [vif_TV, vif_radio, vif_newspaper]}

# vif_frame = pd.DataFrame(d1)

# vif_frame
```

2nd way : VIF

Out

	variable	VIF
0	TV	1.004611
1	Radio	1.144952
2	newspaper	1.145187

If the VIF model > (greater) 4. For any independent variable, drop that particular variable

Rishu
12/04/22
4:40 pm

3) Dt: 16/9/22

$[SSE)_{min}$

$[E(y-\hat{y})^2]_{min}$

$\Rightarrow$ AV plot [Added Variable plot)

$\frac{d(o)}{dx}$ at $x=0$

Partial differention instead of normal differentition. → Apply $\frac{dou}{dau(x}$ $\Rightarrow$ $\frac{\partial}{\partial x}$

#

What is partial differention ?

$\sum [y-\hat{y}]^2$ → it should be Minimum

$\{[y-[\beta_0+\beta_1 x_1+\beta_2 x_2+\beta_3 x_3]]^2$ minimum

$\Sigma[y-\beta_0-\beta_1 x-\beta_2 x_2+\beta_3 x_3]]^2$ minimum

In Simple linear Regression ?

$\frac{d}{dx}[E[y-\beta_0-\beta_1 x_1)^2]$ at $x=0$ → He we have only one variable.

In Multiple linear Regression ?

Constant Calculate Constant

$\frac{\partial}{\partial x}\Sigma[y-\beta_0-\underline{\beta_1 x}-\underline{\beta_2 x_2}-\underline{\beta_3 x_3}]]^2$

calculate Constant constant

↓

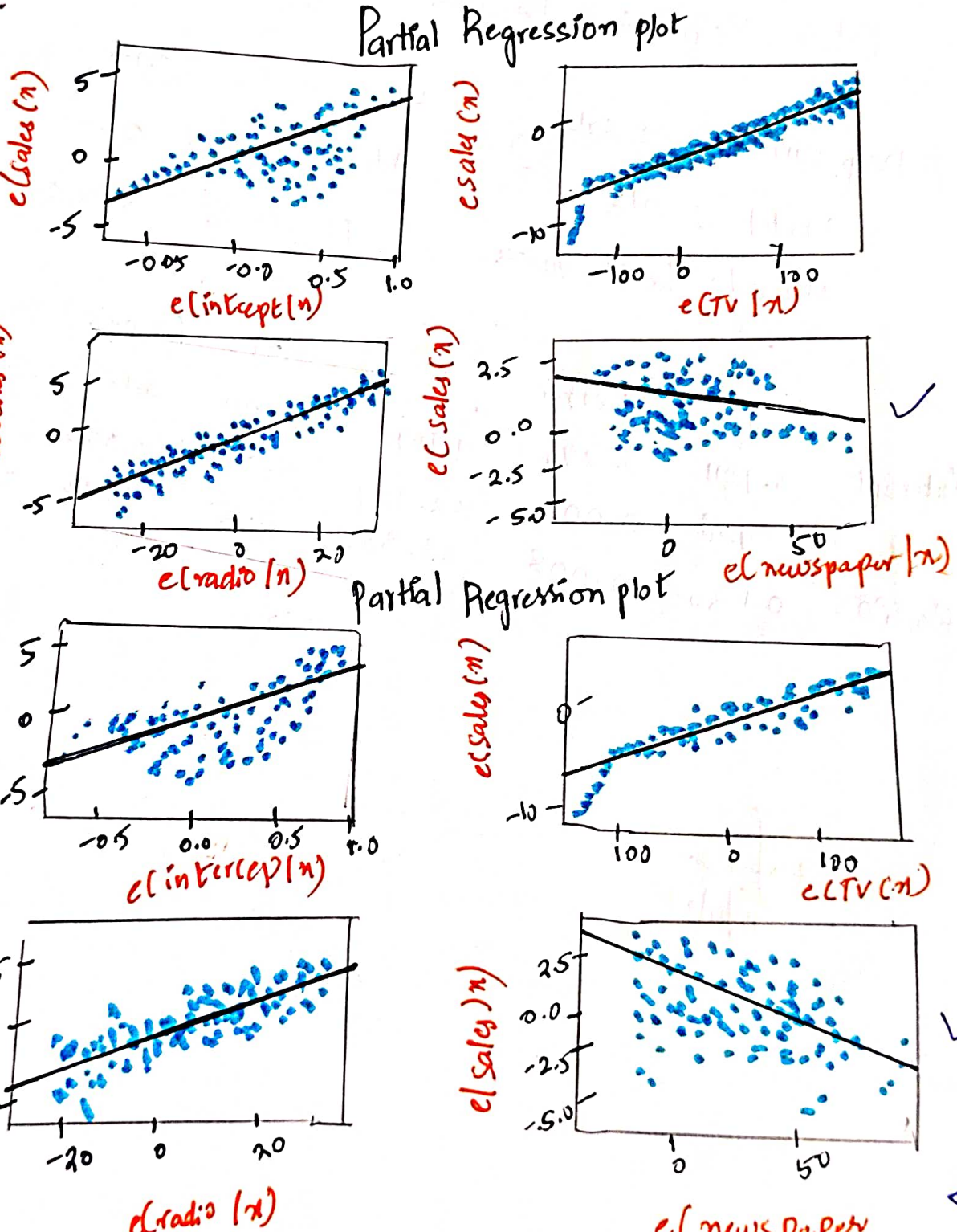
when we calculate with $\frac{dou}{dau(x)}$ $\frac{\partial}{\partial x}$.

When we calculate with one variable other variable will be Constant.

⇒ it is apply Simple Linear Regression on x_1 and x_2 Simple Linear Reg on x_3. it is Applying individually S.L.R on Each and Every individual. variable.

sm.graphics.plot_(partregress)- grid (lm)
Partregress

Out:



Added variable plot is not showing any significance For

⇒ Final model including "TV" and "Radio" Only

```
# final_model = smf.ols(formula = "Sales ~TV + radio",
                        data = df).fit()

# final_model.Summary()
```

Out

OLS Regression Results

Dep. Variable : sales
Model : OLS
Method : Least Squares

R-squared : 0.897 = 90%
Adj. R-squared : 0.896
F-statics = 859.6

	Coeff	Std. error	t	P(>|t|)	(0.025)	(0.975)
Intercept	2.9211	0.294	9.919	0.000	2.340	3.502
TV	0.0458	0.001	32.909	0.000	0.043	0.048
Radio	0.1880	0.008	23.382	0.00	0.172	0.204

16/4/22