# Project – Advertisement Budget

In [1]:

```
import numpy as py
import pandas as pd
```

In [2]:

```
Adv=pd.read_csv("Advertising1.csv")
```

In [3]:

```
Adv.head()
```

Out[3]:

| | Unnamed: 0 | TV | Radio | Newspaper | Sales |
|---|---|---|---|---|---|
| **0** | 1 | 230.1 | 37.8 | 69.2 | 22.1 |
| **1** | 2 | 44.5 | 39.3 | 45.1 | 10.4 |
| **2** | 3 | 17.2 | 45.9 | 69.3 | 9.3 |
| **3** | 4 | 151.5 | 41.3 | 58.5 | 18.5 |
| **4** | 5 | 180.8 | 10.8 | 58.4 | 12.9 |

In [4]:

```
Adv.tail()
```

Out[4]:

| | Unnamed: 0 | TV | Radio | Newspaper | Sales |
|---|---|---|---|---|---|
| **195** | 196 | 38.2 | 3.7 | 13.8 | 7.6 |
| **196** | 197 | 94.2 | 4.9 | 8.1 | 9.7 |
| **197** | 198 | 177.0 | 9.3 | 6.4 | 12.8 |
| **198** | 199 | 283.6 | 42.0 | 66.2 | 25.5 |
| **199** | 200 | 232.1 | 8.6 | 8.7 | 13.4 |

In [5]:

```
#Statastical functions
Adv.describe()
```

Out[5]:

| | Unnamed: 0 | TV | Radio | Newspaper | Sales |
|---|---|---|---|---|---|
| **count** | 200.000000 | 200.000000 | 200.000000 | 200.000000 | 200.000000 |
| **mean** | 100.500000 | 147.042500 | 23.264000 | 30.554000 | 14.022500 |
| **std** | 57.879185 | 85.854236 | 14.846809 | 21.778621 | 5.217457 |
| **min** | 1.000000 | 0.700000 | 0.000000 | 0.300000 | 1.600000 |
| **25%** | 50.750000 | 74.375000 | 9.975000 | 12.750000 | 10.375000 |
| **50%** | 100.500000 | 149.750000 | 22.900000 | 25.750000 | 12.900000 |
| **75%** | 150.250000 | 218.825000 | 36.525000 | 45.100000 | 17.400000 |
| **max** | 200.000000 | 296.400000 | 49.600000 | 114.000000 | 27.000000 |

In [6]:
```
#Presence of null values
Adv.isnull().sum()
```

Out[6]:
```
Unnamed: 0    0
TV            0
Radio         0
Newspaper     0
Sales         0
dtype: int64
```

In [7]:
```
#Data Types of the attributes
Adv.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200 entries, 0 to 199
Data columns (total 5 columns):
 #   Column      Non-Null Count  Dtype  
---  ------      --------------  -----  
 0   Unnamed: 0  200 non-null    int64  
 1   TV          200 non-null    float64
 2   Radio       200 non-null    float64
 3   Newspaper   200 non-null    float64
 4   Sales       200 non-null    float64
dtypes: float64(4), int64(1)
memory usage: 7.9 KB
```

In [8]:
```
#Output is numerical then Linear Regression are Applied

#Important:
#Project Steps Followed:
    #Define Project Goals/Objective
    #Data Retrieval
    #Data Cleansing
    #Exploratory Data Analysis
    #Data Modeling
    #Result Analyis
```

In [9]:
```
#Show diamentions of Data
Adv.shape
```

Out[9]:
```
(200, 5)
```

In [10]:
```
import matplotlib.pyplot as plt
import seaborn as sns
```

In [11]:
```
#Radio minimum value is 0 so is true then ans is 1

(Adv==0).sum(axis=0)
```

Out[11]:
```
Unnamed: 0    0
TV            0
Radio         1
Newspaper     0
Sales         0
dtype: int64
```

# Response Variable Analysis

In [12]:
```
Adv.Sales.value_counts()
```

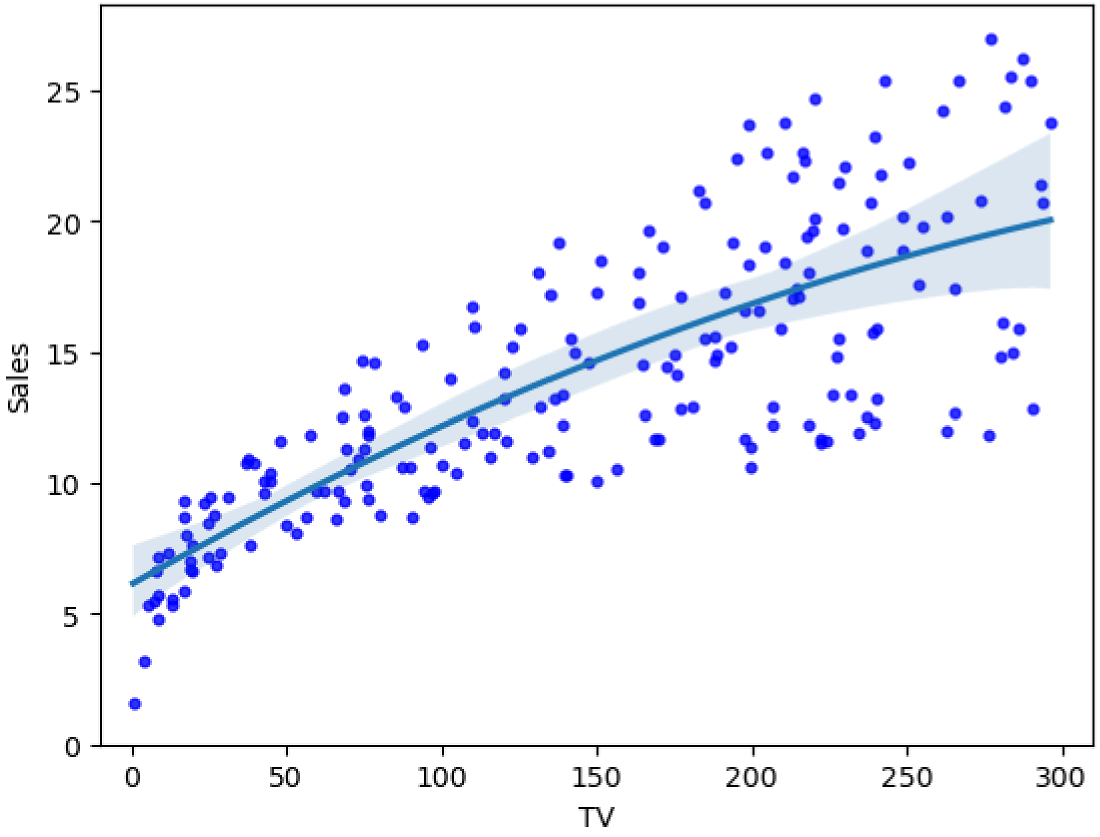
Out[12]:
```
Sales
9.7     5
11.7    4
12.9    4
15.9    4
20.7    3
       ..
17.0    1
18.3    1
22.3    1
14.0    1
25.5    1
Name: count, Length: 121, dtype: int64
```

# Relation between Sales and TV

In [13]:
```
sns.regplot(x=Adv.TV,y=Adv.Sales,order=2,ci=100,scatter_kws={'color':'b','s

plt.xlim(-10,310)
plt.ylim(bottom=0)
plt.show()

#order 1 for linear model (degree)(X^1)
#ci-confidence interval (None / 95 / 99) Range
#scatter_kws Color-red size-9
```

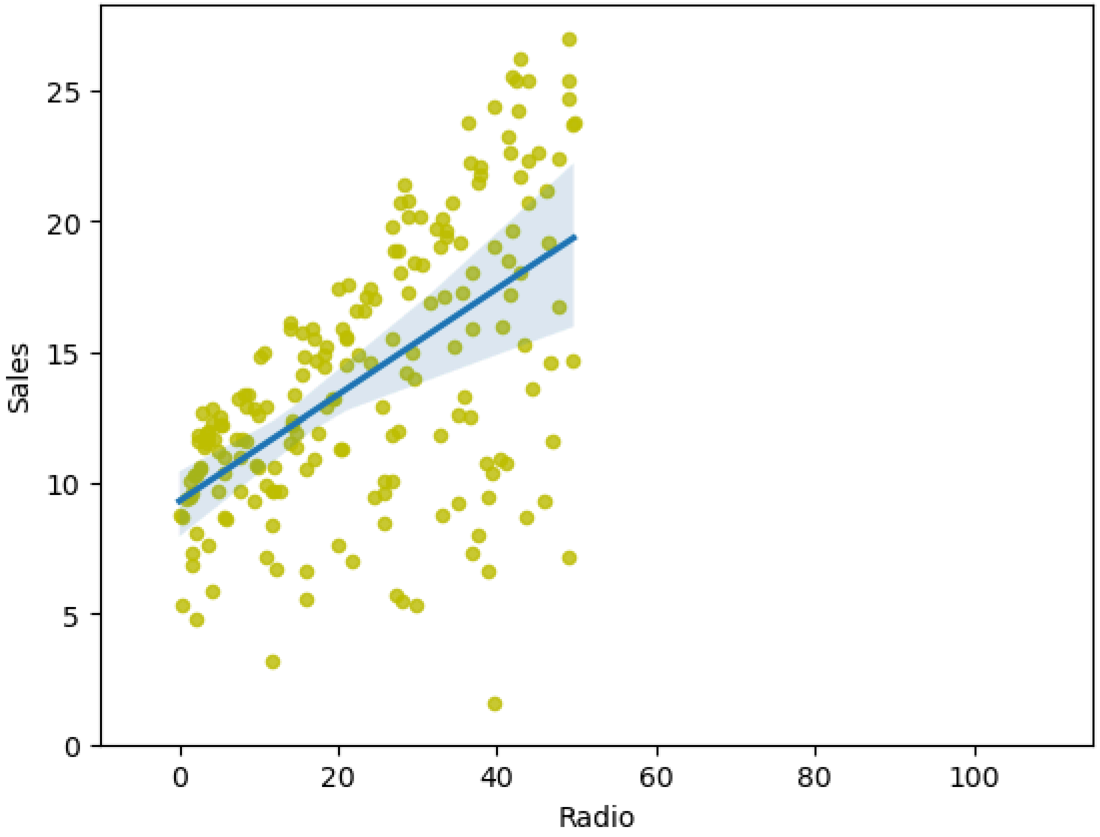


# Relation between Sales and Radio

In [14]:

```
sns.regplot(x=Adv.Radio,y=Adv.Sales,order=1,ci=100,scatter_kws={'color':'y'

plt.xlim(-10,115)
plt.ylim(bottom=0)
plt.show()
```

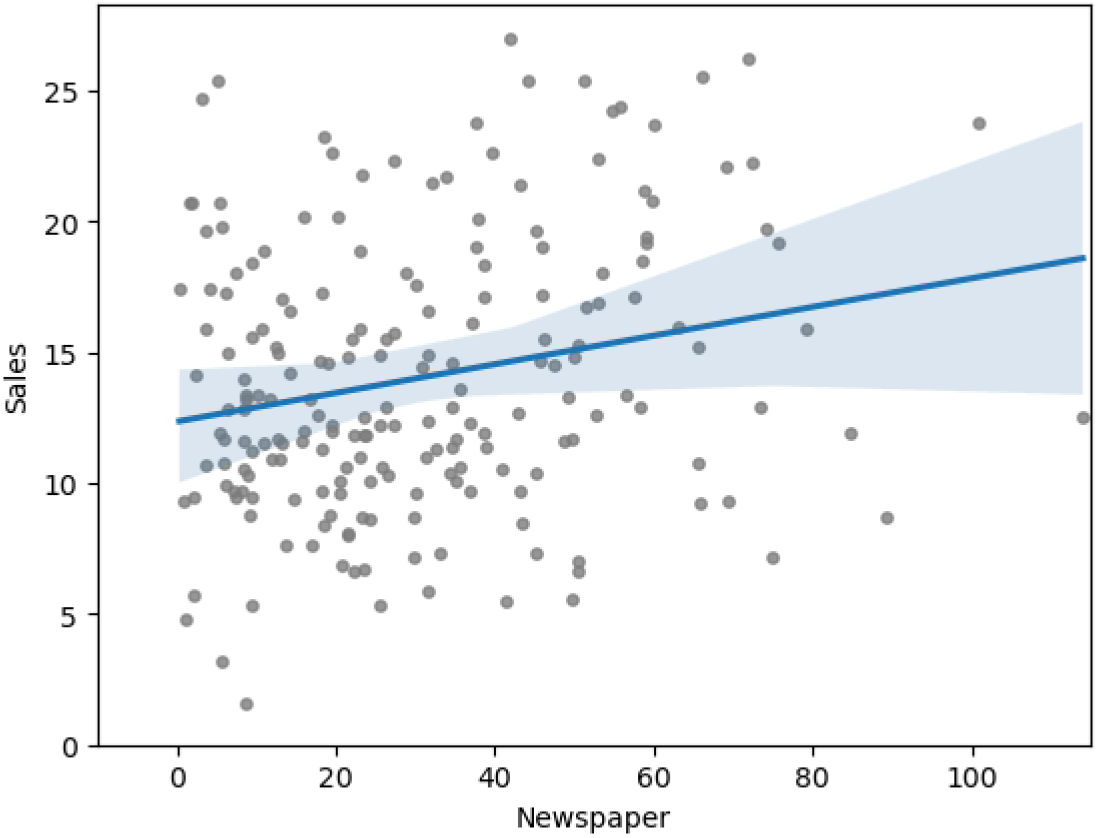


# Relation between Sales and Newspaper

In [15]:

```
sns.regplot(x=Adv.Newspaper,y=Adv.Sales,order=1,ci=100,scatter_kws={'color'

plt.xlim(-10,115)
plt.ylim(bottom=0)
plt.show()
```



# Regression using sklearn

# TV or Sales Relationship

In [30]:

```
import sklearn.linear_model as skl_lm

regr = skl_lm.LinearRegression()

X=Adv.TV.values.reshape(-1,1)

y=Adv.Sales

regr.fit(X,y)
```

Out[30]:

```
▾ LinearRegression
LinearRegression()
```

In [31]:

```
#beta 0
regr.intercept_
```

Out[31]:

```
7.032593549127694
```

In [32]:

```
#beta 1
regr.coef_
```

Out[32]:

```
array([0.04753664])
```

# RSS & MSE

In [33]:

```
#Residual Sum of Square
min_rss=py.sum((regr.intercept_+regr.coef_*X-y.values.reshape(-1,1))**2)

min_rss
```

Out[33]:

```
2102.5305831313517
```

In [34]:

```
#Mean square Eroor
mse=min_rss/len(y)
```

In [35]:

```
mse
```

Out[35]:

```
10.512652915656759
```

# MSE, R-Sq Using Sklearn

In [22]:

```
from sklearn.metrics import mean_squared_error,r2_score
```

In [23]:

```
Sales_pred=regr.predict(X)

r2_score(y,Sales_pred)
```

Out[23]:

```
0.611875050850071
```

In [24]:

```
mean_squared_error(y,Sales_pred)
```

Out[24]:

```
10.512652915656759
```

# Regression Summary using statsmodels

In [25]:
```
import statsmodels.formula.api as smf


est=smf.ols('Sales ~ TV',Adv).fit()

est.summary()
```

Out[25]:

OLS Regression Results

| | | | |
|---|---|---|---|
| **Dep. Variable:** | Sales | **R-squared:** | 0.612 |
| **Model:** | OLS | **Adj. R-squared:** | 0.610 |
| **Method:** | Least Squares | **F-statistic:** | 312.1 |
| **Date:** | Wed, 28 Feb 2024 | **Prob (F-statistic):** | 1.47e-42 |
| **Time:** | 18:46:18 | **Log-Likelihood:** | -519.05 |
| **No. Observations:** | 200 | **AIC:** | 1042. |
| **Df Residuals:** | 198 | **BIC:** | 1049. |
| **Df Model:** | 1 | | |
| **Covariance Type:** | nonrobust | | |

| | **coef** | **std err** | **t** | **P>\|t\|** | **[0.025** | **0.975]** |
|---|---|---|---|---|---|---|
| **Intercept** | 7.0326 | 0.458 | 15.360 | 0.000 | 6.130 | 7.935 |
| **TV** | 0.0475 | 0.003 | 17.668 | 0.000 | 0.042 | 0.053 |

| | | | |
|---|---|---|---|
| **Omnibus:** | 0.531 | **Durbin-Watson:** | 1.935 |
| **Prob(Omnibus):** | 0.767 | **Jarque-Bera (JB):** | 0.669 |
| **Skew:** | -0.089 | **Prob(JB):** | 0.716 |
| **Kurtosis:** | 2.779 | **Cond. No.** | 338. |

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

# Regression RSS, MSE Using statsmodels

In [26]:
```
est.params
```

Out[26]:
```
Intercept    7.032594
TV           0.047537
dtype: float64
```

# RSS

In [28]:
```
((Adv.Sales - (est.params[0] + est.params[1]* Adv.TV))** 2).sum()
```

Out[28]:
```
2102.530583131351
```

# MSE

In [29]:

```
 ((Adv.Sales - (est.params[0] + est.params[1]*Adv.TV))** 2).sum()/len(Adv.S
```

Out[29]:

```
10.512652915656753
```

# Linear Regression for Radio

In [36]:

```
est = smf.ols('Sales ~ Radio', Adv).fit()
print(est.summary().tables[1])
```

```
==========================================================================
====
                 coef    std err          t      P>|t|      [0.025      0.
975]
--------------------------------------------------------------------------
----
Intercept      9.3116      0.563     16.542      0.000       8.202       1
0.422
Radio          0.2025      0.020      9.921      0.000       0.162       
0.243
==========================================================================
====
```

# Linear Regression for Newspaper

In [37]:

```
est = smf.ols('Sales ~ Newspaper', Adv).fit()
print(est.summary().tables[1])
```

```
==========================================================================
====
                 coef    std err          t      P>|t|      [0.025      0.
975]
--------------------------------------------------------------------------
----
Intercept     12.3514      0.621     19.876      0.000      11.126       1
3.577
Newspaper      0.0547      0.017      3.300      0.001       0.022       
0.087
==========================================================================
====
```

# Multiple Linear Regression

In [39]:

```
est = smf.ols('Sales ~ TV + Radio + Newspaper',Adv).fit()

est.summary()
print(est.summary().tables[1])
```

```
==========================================================================
====
                 coef    std err          t      P>|t|      [0.025      0.
975]
--------------------------------------------------------------------------
----
Intercept      2.9389      0.312      9.422      0.000       2.324
3.554
TV             0.0458      0.001     32.809      0.000       0.043
0.049
Radio          0.1885      0.009     21.893      0.000       0.172
0.206
Newspaper     -0.0010      0.006     -0.177      0.860      -0.013
0.011
==========================================================================
====
```

# Correlation

In [40]:

```
Adv.corr()
```

Out[40]:

| | Unnamed: 0 | TV | Radio | Newspaper | Sales |
|---|---|---|---|---|---|
| **Unnamed: 0** | 1.000000 | 0.017715 | -0.110680 | -0.154944 | -0.051616 |
| **TV** | 0.017715 | 1.000000 | 0.054809 | 0.056648 | 0.782224 |
| **Radio** | -0.110680 | 0.054809 | 1.000000 | 0.354104 | 0.576223 |
| **Newspaper** | -0.154944 | 0.056648 | 0.354104 | 1.000000 | 0.228299 |
| **Sales** | -0.051616 | 0.782224 | 0.576223 | 0.228299 | 1.000000 |

In [ ]: