In [1]:

```python
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import os
import statsmodels.formula.api as sm
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error
from sklearn.model_selection import train_test_split
import warnings
import statsmodels.api as sm
```

In [2]:

```python
df=pd.read_csv("C:/Users/hp/Downloads/archive (6)/Advertising.csv")
```

In [3]:

```python
df.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]:

```python
df.columns
```

Out[4]:

```
Index(['Unnamed: 0', 'TV', 'Radio', 'Newspaper', 'Sales'], dtype='object')
```

In [5]:

```python
df.rename(columns={'Unnamed: 0':'Index'},inplace=True)
```

In [6]:
```
df
```

Out[6]:

	Index	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
...	...	...	...	...	...
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

200 rows × 5 columns

In [7]:
```
df.info
```

Out[7]:
```
<bound method DataFrame.info of      Index     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
..     ...    ...    ...        ...    ...
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

[200 rows x 5 columns]>
```

In [8]:
```
df.describe().T
```

Out[8]:

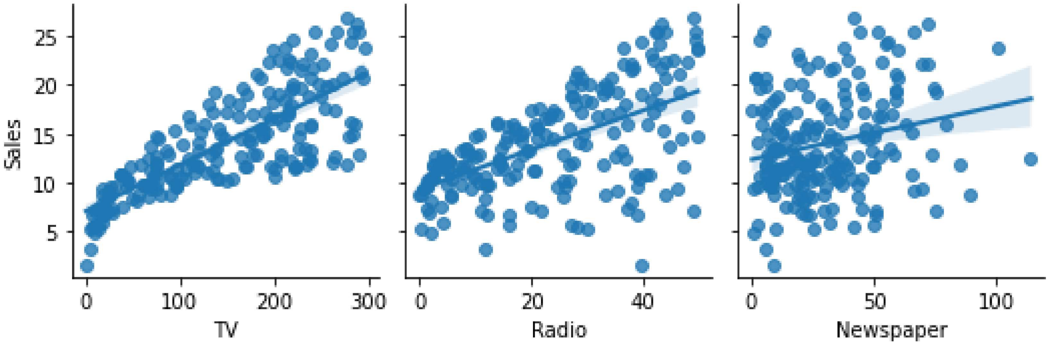
	count	mean	std	min	25%	50%	75%	max
Index	200.0	100.5000	57.879185	1.0	50.750	100.50	150.250	200.0
TV	200.0	147.0425	85.854236	0.7	74.375	149.75	218.825	296.4
Radio	200.0	23.2640	14.846809	0.0	9.975	22.90	36.525	49.6
Newspaper	200.0	30.5540	21.778621	0.3	12.750	25.75	45.100	114.0
Sales	200.0	14.0225	5.217457	1.6	10.375	12.90	17.400	27.0

In [9]:
```
df.isnull().values.any()
df.isnull().sum()
```

Out[9]:
```
Index        0
TV           0
Radio        0
Newspaper    0
Sales        0
dtype: int64
```

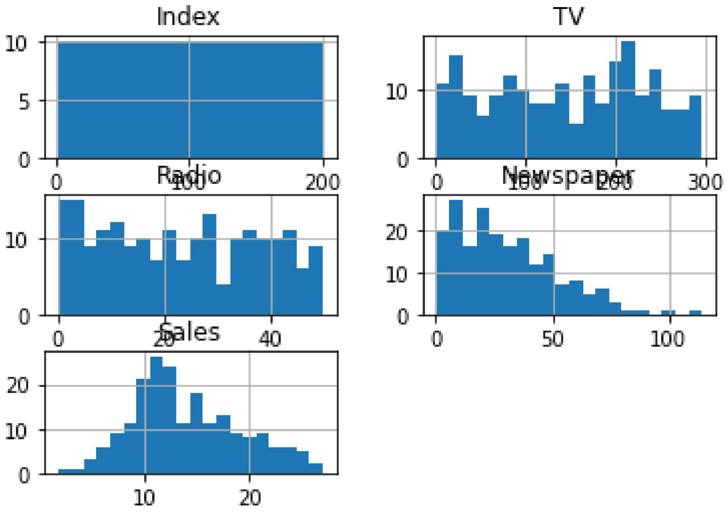
In [10]:
```
sns.pairplot(df,x_vars=["TV","Radio","Newspaper"],y_vars="Sales",kind="reg")
```

Out[10]:
```
<seaborn.axisgrid.PairGrid at 0x1ce37ba0f10>
```



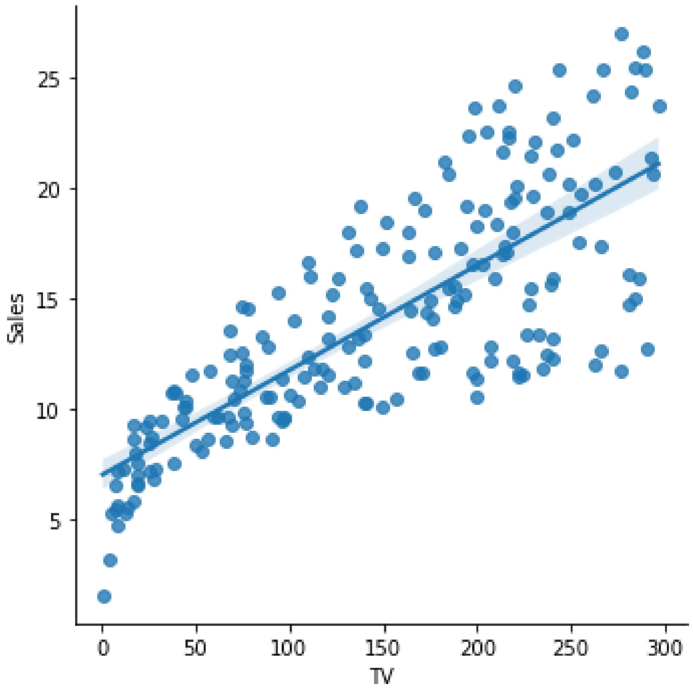
```
In [11]: df.hist(bins=20)
```

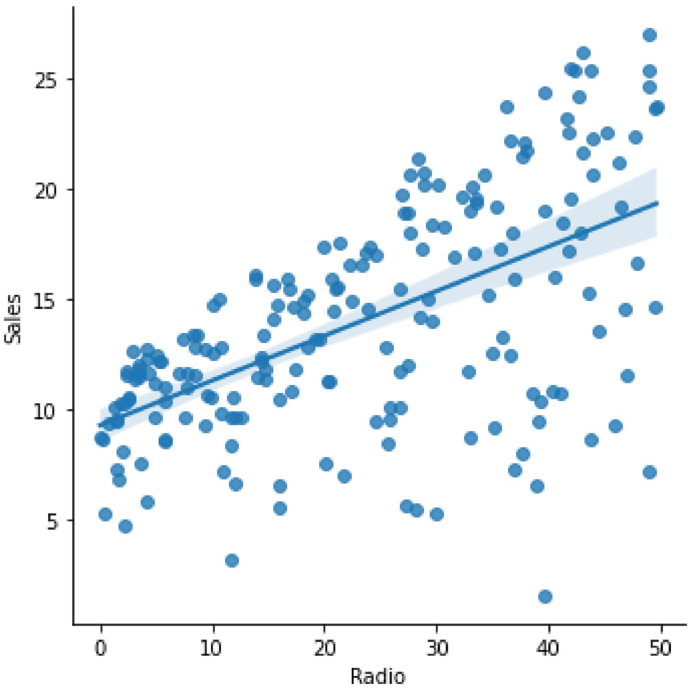
```
Out[11]: array([[<AxesSubplot:title={'center':'Index'}>,
                 <AxesSubplot:title={'center':'TV'}>],
                [<AxesSubplot:title={'center':'Radio'}>,
                 <AxesSubplot:title={'center':'Newspaper'}>],
                [<AxesSubplot:title={'center':'Sales'}>, <AxesSubplot:>]],
               dtype=object)
```

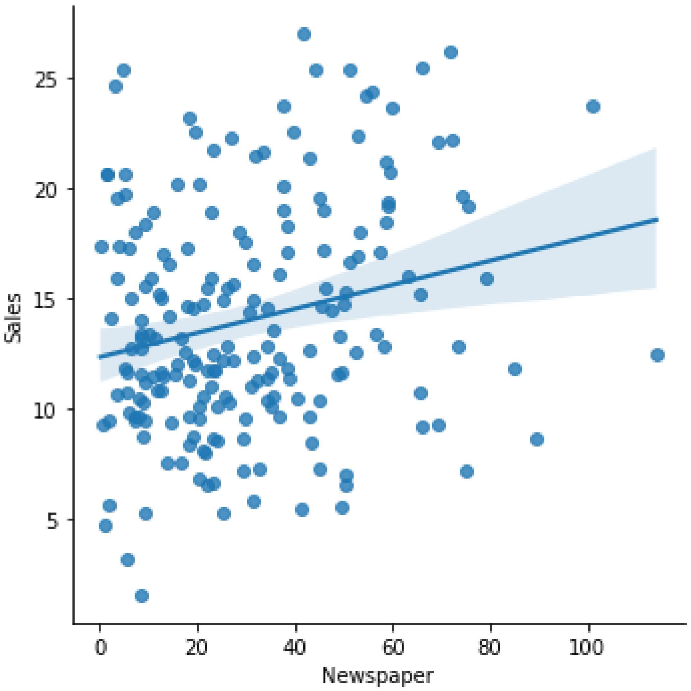


```
In [12]: sns.lmplot(x='TV',y='Sales',data=df)
         sns.lmplot(x='Radio',y='Sales',data=df)
         sns.lmplot(x='Newspaper',y='Sales',data=df)
```

```
Out[12]: <seaborn.axisgrid.FacetGrid at 0x1ce3b0ee190>
```




Sales
25
20
15
10
5
0
10
20
30
40
50
Radio

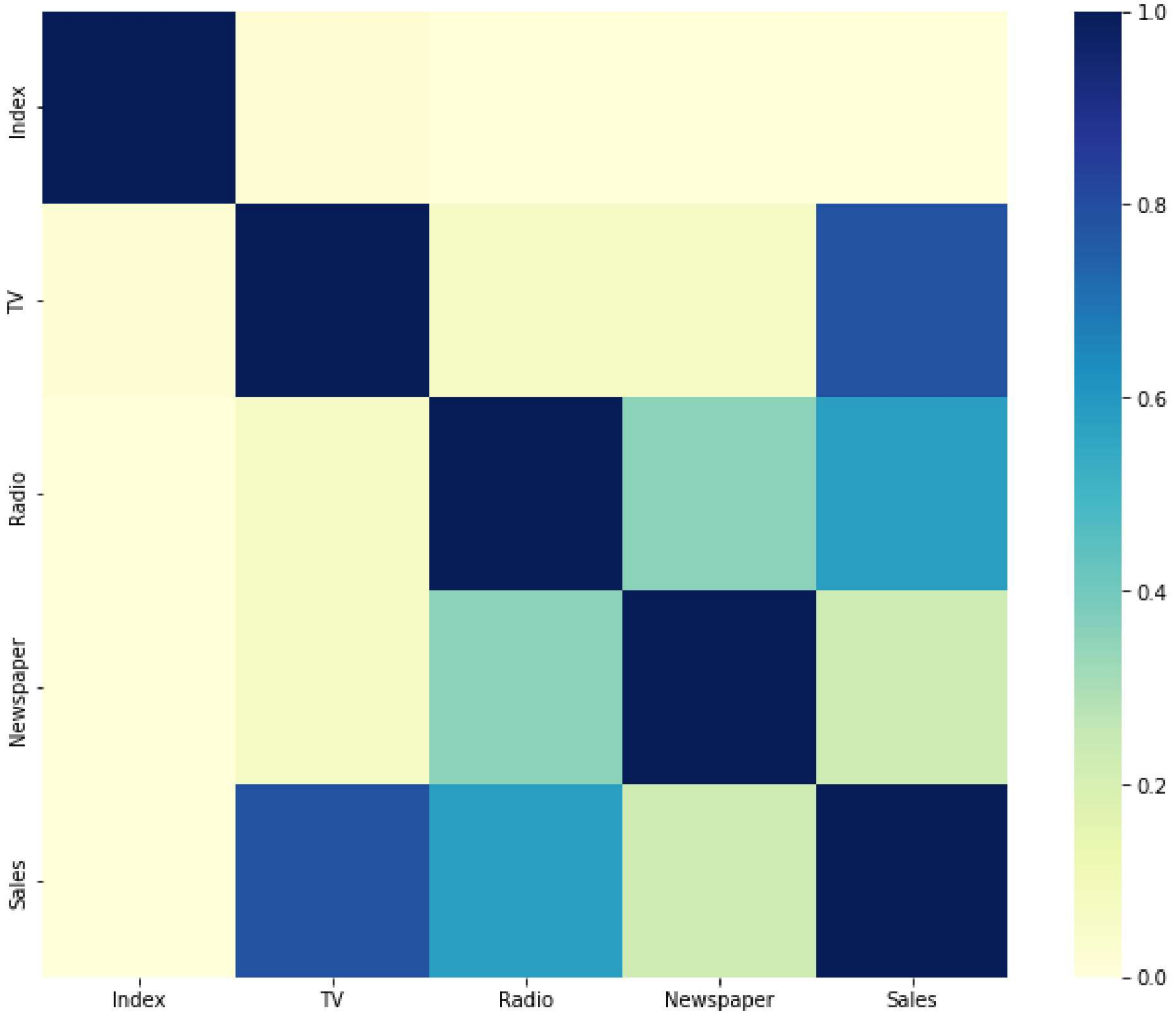

Sales
25
20
15
10
5
0
20
40
60
80
100
Newspaper

In [13]:

```
corrmat=df.corr()
f,ax=plt.subplots(figsize=(12,9))
sns.heatmap(corrmat,vmin=0,vmax=1,square=True,cmap="YlGnBu", ax=ax)
```

Out[13]:

```
<AxesSubplot:>
```



In [14]:
```python
x=df.drop('Sales',axis=1)
y=df[["Sales"]]
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=42)
```

In [15]:
```python
formula="Sales~TV+Radio+Newspaper"
lin_model=sm.OLS.from_formula(formula,data=df).fit()
print(lin_model.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                  Sales   R-squared:                       0.897
Model:                            OLS   Adj. R-squared:                  0.896
Method:                 Least Squares   F-statistic:                     570.3
Date:                Sun, 05 Nov 2023   Prob (F-statistic):           1.58e-96
Time:                        18:47:00   Log-Likelihood:                -386.18
No. Observations:                 200   AIC:                             780.4
Df Residuals:                     196   BIC:                             793.6
Df Model:                           3
Covariance Type:            nonrobust
==============================================================================
                 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
==============================================================================
Omnibus:                       60.414   Durbin-Watson:                   2.084
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              151.241
Skew:                          -1.327   Prob(JB):                     1.44e-33
Kurtosis:                       6.332   Cond. No.                         454.
==============================================================================

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

In [16]:
```
print(lin_model.params,"\n")
```

```
Intercept     2.938889
TV            0.045765
Radio         0.188530
Newspaper    -0.001037
dtype: float64
```

In [17]:
```
results=[]
names=[]
```

In [18]:
```
models=[('LinearRegression',LinearRegression())];models
```

Out[18]:
```
[('LinearRegression', LinearRegression())]
```

In [20]:
```
for name,model in models:
    model.fit(x_train,y_train)
    y_pred=model.predict(x_test)
    result=np.sqrt(mean_squared_error(y_test, y_pred))
    results.append(result)
    names.append(name)
    message="%s: %f"%(name,result)
    print(message)
```

```
LinearRegression: 1.788576
```

In [21]:
```
new_dta=pd.DataFrame({'TV':[50],'Radio':[30],'Newspaper':[25]})
predicted_sales=lin_model.predict(new_dta)
print('Predicted Sales:', predicted_sales)
```

```
Predicted Sales: 0    10.857085
dtype: float64
```

In [22]:

```
new_dta=pd.DataFrame({'TV':[100],'Radio':[80],'Newspaper':[65]})
predicted_sales=lin_model.predict(new_dta)
print('Predicted Sales:', predicted_sales)
```

```
Predicted Sales: 0    22.530318
dtype: float64
```

In [ ]: