In [4]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt

from scipy import stats
import pylab

import warnings
warnings.filterwarnings('ignore')

pd.set_option("display.max_columns",None)
pd.set_option("display.max_rows",None)

from sklearn.linear_model import LinearRegression
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor,AdaBoostRegressor

from sklearn.model_selection import train_test_split
from sklearn.metrics import r2_score
```

In [7]:
```
df = pd.read_csv('sales.csv')
```

In [8]:
```
df.head()
```

Out[8]:

	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 [9]:
```
df.isnull().sum()
```

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

In [10]:
```
df.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 [11]:
```
print('Duplicate_values=',df.duplicated().sum())
```

```
Duplicate_values= 0
```

In [12]:
```
df.head()
```

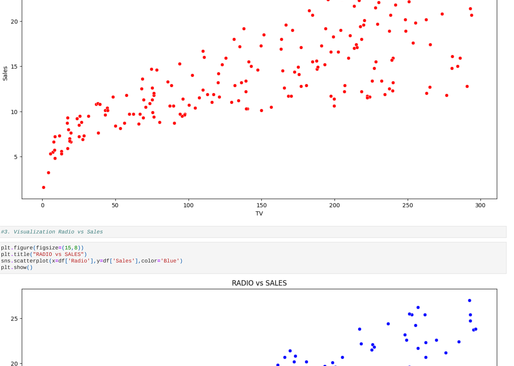
Out[12]:

	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

1. Visualizing targate variable 'sales'

In [14]:
```
plt.figure(figsize=(20,8))
plt.subplot(1,2,1)
plt.title('Sales Distribution')
sns.histplot(data=df,x='Sales',kde=True,color='red')

plt.subplot(1,2,2)
plt.title('spread of sale')
sns.boxenplot(data=df,x='Sales',color='purple')
plt.show()
```



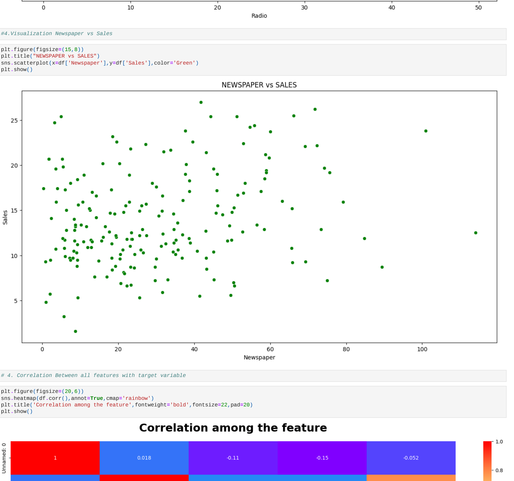
insights

most of the value 8-29

formed bell shape curve

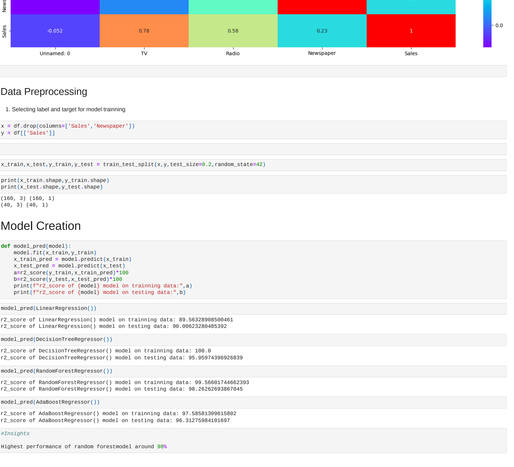
In [15]:
```
#2.Visualize Tv vs Sales features
```

In [16]:
```
plt.figure(figsize=(15,8))
plt.title("TV vs SALES")
sns.scatterplot(x=df['TV'],y=df['Sales'],color='red')
plt.show()
```



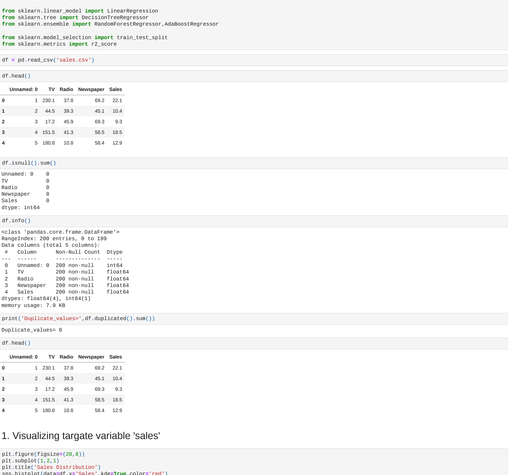
In [19]:
```
#3. Visualization Radio vs Sales
```

In [20]:
```
plt.figure(figsize=(15,8))
plt.title("RADIO vs SALES")
sns.scatterplot(x=df['Radio'],y=df['Sales'],color='Blue')
plt.show()
```



In [21]:
```
#4.Visualization Newspaper vs Sales
```

In [22]:
```
plt.figure(figsize=(15,8))
plt.title("NEWSPAPER vs SALES")
sns.scatterplot(x=df['Newspaper'],y=df['Sales'],color='Green')
plt.show()
```



In [23]:
```
# 4. Correlation Between all features with target variable
```

In [27]:
```
plt.figure(figsize=(20,6))
sns.heatmap(df.corr(),annot=True,cmap='rainbow')
plt.title('Correlation among the feature',fontweight='bold',fontsize=22,pad=20)
plt.show()
```

In [ ]:
```
```

Data Preprocessing

1. Selecting label and target for model trainning

In [34]:
```
x = df.drop(columns=['Sales','Newspaper'])
y = df[['Sales']]
```

In [ ]:
```
```

In [37]:
```
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.2,random_state=42)
```

In [38]:
```
print(x_train.shape,y_train.shape)
print(x_test.shape,y_test.shape)
```

```
(160, 3) (160, 1)
(40, 3) (40, 1)
```

Model Creation

In [41]:
```
def model_pred(model):
    model.fit(x_train,y_train)
    x_train_pred = model.predict(x_train)
    x_test_pred = model.predict(x_test)
    a=r2_score(y_train,x_train_pred)*100
    b=r2_score(y_test,x_test_pred)*100
    print(f"r2_score of {model} model on training data:",a)
    print(f"r2_score of {model} model on testing data:",b)
```

In [42]:
```
model_pred(LinearRegression())
```

```
r2_score of LinearRegression() model on training data: 89.56328906580461
r2_score of LinearRegression() model on testing data: 90.60623280485392
```

In [43]:
```
model_pred(DecisionTreeRegressor())
```

```
r2_score of DecisionTreeRegressor() model on training data: 100.0
r2_score of DecisionTreeRegressor() model on testing data: 95.95974396926839
```

In [44]:
```
model_pred(RandomForestRegressor())
```

```
r2_score of RandomForestRegressor() model on trainning data: 99.56601744662393
r2_score of RandomForestRegressor() model on testing data: 98.26262093867045
```

In [45]:
```
model_pred(AdaBoostRegressor())
```

```
r2_score of AdaBoostRegressor() model on trainning data: 97.58581309615802
r2_score of AdaBoostRegressor() model on testing data: 96.31275984191697
```

In [ ]:
```
#Insights

Highest performance of random forestmodel around 98%
```