```
import pandas as pd
import numpy as np
import seaborn as sns
from scipy.stats import skew
%matplotlib inline
```

```
import matplotlib.pyplot as plt
plt.style.use("ggplot")
plt.rcParams['figure.figsize'] = (12, 8)
```

```
df=pd.read_csv('Advertising.csv')
```

```
df.head(10)
```

	TV	radio	newspaper	sales
0	230.1	37.8	69.2	22.1
1	44.5	39.3	45.1	10.4
2	17.2	45.9	69.3	9.3
3	151.5	41.3	58.5	18.5
4	180.8	10.8	58.4	12.9
5	8.7	48.9	75.0	7.2
6	57.5	32.8	23.5	11.8
7	120.2	19.6	11.6	13.2
8	8.6	2.1	1.0	4.8
9	199.8	2.6	21.2	10.6

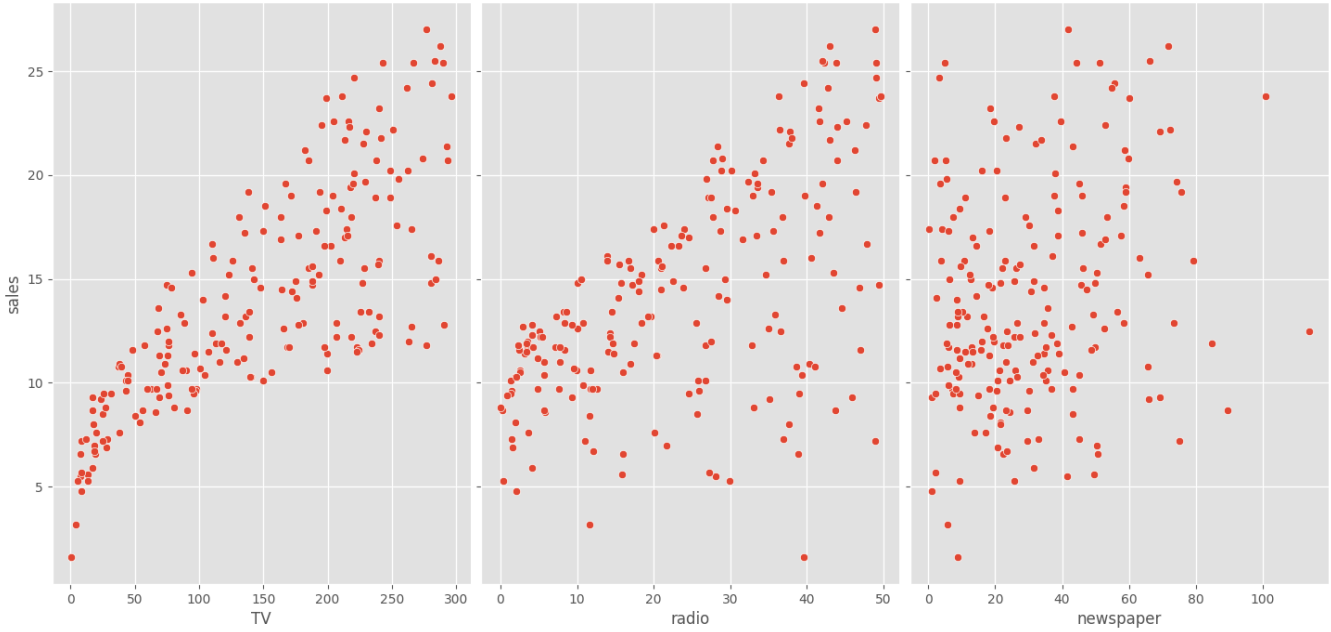
Next steps: Generate code with df | View recommended plots | New interactive sheet

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200 entries, 0 to 199
Data columns (total 4 columns):
 #   Column     Non-Null Count  Dtype  
---  ------     --------------  -----  
 0   TV         200 non-null    float64
 1   radio      200 non-null    float64
 2   newspaper  200 non-null    float64
 3   sales      200 non-null    float64
dtypes: float64(4)
memory usage: 6.4 KB
```

```
sns.pairplot(df,x_vars=['TV','radio','newspaper'],y_vars='sales',height=7,aspect=0.7)
```

```
<seaborn.axisgrid.PairGrid at 0x7d3e65ccdc60>
```




```
from sklearn.linear_model import LinearRegression
x=df.drop('sales',axis=1)
y=df['sales']
lm1=LinearRegression()
lm1.fit(x,y)
print(lm1.intercept_)
print(lm1.coef_)
```

```
2.938889369459412
[ 0.04576465  0.18853002 -0.00103749]
```

```
list(zip(['TV','radio','newspaper'],lm1.coef_))
```

```
[('TV', 0.0457646454553976),
 ('radio', 0.18853001691820448),
 ('newspaper', -0.0010374930424763285)]
```

```
sns.heatmap(df.corr(),annot=True)
```

<Axes: >


```
from sklearn.metrics import r2_score
lm2=LinearRegression()
lm2.fit(x[['TV','radio']],y)
lm2_pred=lm2.predict(x[['TV','radio']])
print(r2_score(y,lm2_pred))
```

```
0.8971942610828957
```

```
lm3=LinearRegression()
lm3.fit(x[['TV','radio','newspaper']],y)
lm3_pred=lm3.predict(x[['TV','radio','newspaper']])
print(r2_score(y,lm3_pred))
```

```
0.8972106381789522
```

```
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error
X=df.drop('sales',axis=1)
y=df['sales']
x_train,x_test,y_train,y_test=train_test_split(X,y,random_state=1)

lm4=LinearRegression().fit(x_train,y_train)
lm4_pred=lm4.predict(x_test)
print("RMSE:",np.sqrt(mean_squared_error(y_test,lm4_pred)))
print("R*2:",r2_score(y_test,lm4_pred))
```

```
RMSE: 1.404651423032895
R*2: 0.9156213613792232
```

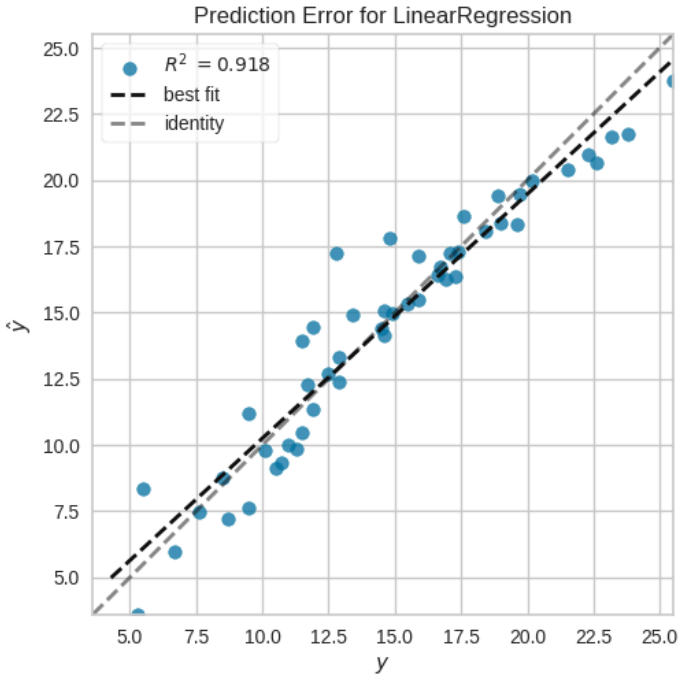
```
X=df.drop(['sales','newspaper'],axis=1)
y=df['sales']
x_train,x_test,y_train,y_test=train_test_split(X,y,random_state=1)

lm5=LinearRegression().fit(x_train,y_train)
lm5_pred=lm5.predict(x_test)
print("RMSE:",np.sqrt(mean_squared_error(y_test,lm5_pred)))
print("R*2:",r2_score(y_test,lm5_pred))
```

```
RMSE: 1.3879034699382888
R*2: 0.9176214942248907
```

```
from yellowbrick.regressor import PredictionError,ResidualsPlot
v=PredictionError(lm5).fit(x_train,y_train)
v.score(x_test,y_test)
v.poof()
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/base.py:465: UserWarning: X does not have valid feature names, but L
  warnings.warn(
```




```
<Axes: title={'center': 'Prediction Error for LinearRegression'}, xlabel='$y$', ylabel='$\\hat{y}$'>
```

```
df['interaction']=df['TV']*df['radio']
X=df[['TV','radio','interaction']]
y=df['sales']
x_train,x_test,y_train,y_test=train_test_split(X,y,random_state=1)

lm6=LinearRegression().fit(x_train,y_train)
lm6_pred=lm6.predict(x_test)
print("RMSE:",np.sqrt(mean_squared_error(y_test,lm6_pred)))
print("R*2:",r2_score(y_test,lm6_pred))
```

```
RMSE: 0.7011871137164328
R*2: 0.978973681468126
```