GRIP: The Sparks Foundation

Data Science and Business Analytics Intern

Name: Nitha Royals GR

Task 1 :Prediction Using Supervised ML

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import warnings
import seaborn as sns
```

In [2]:
```
url = "http://bit.ly/w-data"
data= pd.read_csv(url)
```

In [3]:
```
print(data.shape)
data.head()
```

```
(25, 2)
```

Out[3]:

	Hours	Scores
0	2.5	21
1	5.1	47
2	3.2	27
3	8.5	75
4	3.5	30

In [4]:
```
data.describe()
```

Out[4]:

	Hours	Scores
count	25.000000	25.000000
mean	5.012000	51.480000
std	2.525094	25.286887
min	1.100000	17.000000
25%	2.700000	30.000000
50%	4.800000	47.000000
75%	7.400000	75.000000
max	9.200000	95.000000

In [5]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 25 entries, 0 to 24
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype  
---  ------  --------------  -----  
 0   Hours   25 non-null     float64
 1   Scores  25 non-null     int64  
dtypes: float64(1), int64(1)
memory usage: 528.0 bytes
```

In [6]:
```
data.corr(method='pearson')
```

Out[6]:

	Hours	Scores
Hours	1.000000	0.976191
Scores	0.976191	1.000000

In [7]:
```
data.corr(method='spearman')
```

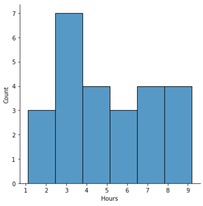
Out[7]:

	Hours	Scores
Hours	1.000000	0.971891
Scores	0.971891	1.000000

In [8]:
```
hours=data['Hours']
scores=data['Scores']
```

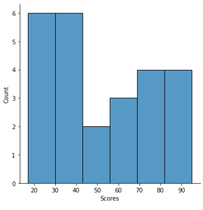
In [9]:
```
sns.displot(hours)
```

Out[9]:
```
<seaborn.axisgrid.FacetGrid at 0x20f374187f0>
```



In [10]:
```
sns.displot(scores)
```

Out[10]:
```
<seaborn.axisgrid.FacetGrid at 0x20f37564f40>
```



Linear Regression

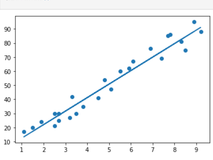
In [11]:
```
x = data.iloc[:, :-1].values
y = data.iloc[:, 1].values
```

In [12]:
```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y,test_size=0.2, random_state=50)
```

In [13]:
```
from sklearn.linear_model import LinearRegression
reg=LinearRegression()
reg.fit(x_train, y_train)
```

Out[13]:
```
LinearRegression()
```

In [14]:
```
m=reg.coef_
c=reg.intercept_
line=m*x+c
plt.scatter(x, y)
plt.plot(x,line);
plt.show()
```



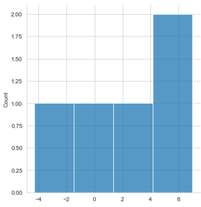
In [15]:
```
y_pred=reg.predict(x_test)
```

In [16]:
```
actual_predicted=pd.DataFrame({'Target':y_test,'predicted':y_pred})
actual_predicted
```

Out[16]:

	Target	predicted
0	95	88.211394
1	30	28.718453
2	76	69.020122
3	35	39.273652
4	17	13.365436

In [17]:
```
sns.set_style('whitegrid')
sns.displot(np.array(y_test-y_pred))
plt.show()
```



In [18]:
```
h=9.25
s=reg.predict([[h]])
print("if a student for {} hpurs per day he/she will score {}% in exam.".format(h,s))
```

```
if a student for 9.25 hpurs per day he/she will score [91.56986604]% in exam.
```

In [19]:
```
from sklearn import metrics
from sklearn.metrics import r2_score
print('Mean Absolute Error:',metrics.mean_absolute_error(y_test,y_pred))
print('R2 score:',r2_score(y_test,y_pred))
```

```
Mean Absolute Error: 4.591649530063028 5
R2 score: 0.9710141413299942
```