Data Science and Business analytics Intern at Spark Foundation #GRIPJAN22

Author :- Dinesh Chandra Gaddam

Task -- 1 :- Prediction Using Supervised ML

Predict the percentage of a student based on the number of study hours.

A simple Linear Regression task that involves two variables

LVL - Beginner

Data found at : http://bit.ly/w-data

---

importing all related libraries..

In [19]:
```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
```

---

Reading data from remote link...

In [21]:
```
url = r"https://raw.githubusercontent.com/AdiPersonalWorks/Random/master/student_scores%20-%20student_scores.cs
s_data = pd.read_csv(url)
print("Data import successful")

s_data.head(12)
```

```
Data import successful
```

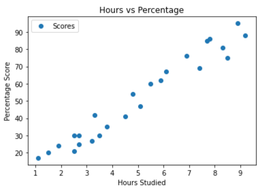
Out[21]:

| | Hours | Scores |
|---|---|---|
| **0** | 2.5 | 21 |
| **1** | 5.1 | 47 |
| **2** | 3.2 | 27 |
| **3** | 8.5 | 75 |
| **4** | 3.5 | 30 |
| **5** | 1.5 | 20 |
| **6** | 9.2 | 88 |
| **7** | 5.5 | 60 |
| **8** | 8.3 | 81 |
| **9** | 2.7 | 25 |
| **10** | 7.7 | 85 |
| **11** | 5.9 | 62 |

---

Plotting the distribution of scores

In [22]:
```
s_data.plot(x='Hours', y='Scores', style='o')
plt.title('Hours vs Percentage')
plt.xlabel('Hours Studied')
plt.ylabel('Percentage Score')
plt.show()
```




In [23]:
```
X = s_data.iloc[:, :-1].values
y = s_data.iloc[:, 1].values

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=0)
regressor = LinearRegression()
regressor.fit(X_train.reshape(-1,1), y_train)

print("Complete")
```

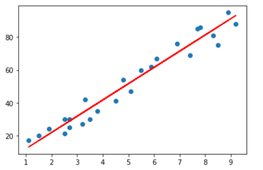
```
Complete
```

---

Plotting the regression line

In [24]:
```
line = regressor.coef_*X+regressor.intercept_

# Plotting for the test data
plt.scatter(X, y)
plt.plot(X, line,color='red');
plt.show()
```



---

Testing data and Model Prediction

In [26]:
```
print(X_test)
y_pred = regressor.predict(X_test)
```

```
[[1.5]
 [3.2]
 [7.4]
 [2.5]
 [5.9]]
```

---

Comparing Actual vs Predicted

In [27]:
```
df = pd.DataFrame({'Actual': y_test, 'Predicted': y_pred})
df
```

Out[27]:

| | Actual | Predicted |
|---|---|---|
| **0** | 20 | 16.884145 |
| **1** | 27 | 33.732261 |
| **2** | 69 | 75.357018 |
| **3** | 30 | 26.794801 |
| **4** | 62 | 60.491033 |

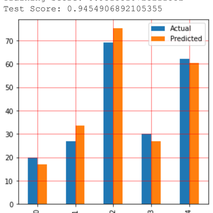
---

Estimating training and test score ,Plotting the Bar graph to depict the difference between the actual and predicted value

In [28]:
```
print("Training Score:",regressor.score(X_train,y_train))
print("Test Score:",regressor.score(X_test,y_test))

df.plot(kind='bar',figsize=(5,5))
plt.grid(which='major', linewidth='0.5', color='red')
plt.grid(which='minor', linewidth='0.5', color='blue')
plt.show()
```

```
Training Score: 0.9515510725211552
Test Score: 0.9454906892105355
```




---

Testing the model with our own data

In [29]:
```
hours = 9.25
test = np.array([hours])
test = test.reshape(-1, 1)
own_pred = regressor.predict(test)
print("No of Hours = {}".format(hours))
print("Predicted Score = {}".format(own_pred[0]))
```

```
No of Hours = 9.25
Predicted Score = 93.69173248737535
```

# Evaluating the model

Here different errors have been calculated to compare the model performance and predict the accuracy.

In [30]:
```
from sklearn import metrics
print('Mean Absolute Error:',metrics.mean_absolute_error(y_test, y_pred))
print('Mean Squared Error:', metrics.mean_squared_error(y_test, y_pred))
print('Root Mean Squared Error:', np.sqrt(metrics.mean_squared_error(y_test, y_pred)))
print('R-2:', metrics.r2_score(y_test, y_pred))
```

```
Mean Absolute Error: 4.183859899002975
Mean Squared Error: 21.598769307217406
Root Mean Squared Error: 4.647447612100367
R-2: 0.9454906892105355
```

## The Accuracy of the Model is 94.55%

## Thank You