Creado por:

Isabel Maniega

Ejercicio 1

Dado el archivo csv Salary_Data.csv, realizar la predicción para este set de datos:

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

In [2]:
```
dataset = pd .read_csv("Salary_Data.csv")
dataset
```

Out[2]:

	YearsExperience	Salary
0	1.1	39343.0
1	1.3	46205.0
2	1.5	37731.0
3	2.0	43525.0
4	2.2	39891.0
5	2.9	56642.0
6	3.0	60150.0
7	3.2	54445.0
8	3.2	64445.0
9	3.7	57189.0
10	3.9	63218.0
11	4.0	55794.0
12	4.0	56957.0
13	4.1	57081.0
14	4.5	61111.0
15	4.9	67938.0
16	5.1	66029.0
17	5.3	83088.0
18	5.9	81363.0
19	6.0	93940.0
20	6.8	91738.0
21	7.1	98273.0
22	7.9	101302.0
23	8.2	113812.0
24	8.7	109431.0
25	9.0	105582.0
26	9.5	116969.0
27	9.6	112635.0
28	10.3	122391.0
29	10.5	121872.0

Sin estandarizar

1) Realiza la predicción para estos datos

In [3]:
```
X = dataset.iloc[:, :-1].values
y = dataset.iloc[:, 1].values
```

In [4]:
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 1/3, random_state = 0)
```

In [5]:
```
from sklearn.linear_model import LinearRegression
regressor = LinearRegression()
regressor.fit(X_train, y_train)
```

Out[5]:
```
LinearRegression()
```

In [6]:
```
# Predicting the Test set results
y_pred = regressor.predict(X_test)
```

In [7]:
```
print('DATOS DEL MODELO REGRESIÓN LINEAL SIMPLE')
print()
print('Valor de la pendiente o coeficiente "a":')
print(regressor.coef_)
print('Valor de la intersección o coeficiente "b":')
print(regressor.intercept_)
print()
print('La ecuación del modelo es igual a:')
print('y = ', regressor.coef_, 'x ', regressor.intercept_)
```

```
DATOS DEL MODELO REGRESIÓN LINEAL SIMPLE

Valor de la pendiente o coeficiente "a":
[9345.94244312]
Valor de la intersección o coeficiente "b":
26816.192244031183

La ecuación del modelo es igual a:
y =  [9345.94244312] x  26816.192244031183
```

In [8]:
```
print("Precisión del modelo:")
print(regressor.score(X_train, y_train))
```

```
Precisión del modelo:
0.9381900012894278
```

2) Realiza dos gráficos para visualizar los datos de Salario frente a Experiencia en "Training set"

In [9]:
```
# Visualising the Training set results
plt.scatter(X_train, y_train, color = 'red')
plt.plot(X_train, regressor.predict(X_train), color = 'blue')
plt.title('Salary vs Experience (Training set)')
plt.xlabel('Years of Experience')
plt.ylabel('Salary')
plt.show()
```

3) Realiza dos gráficos para visualizar los datos de Salario frente a Experiencia en "Test set"

In [10]:
```
# Visualising the Test set results
plt.scatter(X_test, y_test, color = 'red')
plt.plot(X_train, regressor.predict(X_train), color = 'blue')
plt.title('Salary vs Experience (Test set)')
plt.xlabel('Years of Experience')
plt.ylabel('Salary')
plt.show()
```

Con estandarización

1) Realiza la predicción para estos datos

In [11]:
```
X = dataset.iloc[:, :-1].values.reshape(-1, 1)
y = dataset.iloc[:, 1].values.reshape(-1, 1)
```

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 = 1/3, random_state = 0)
```

In [13]:
```
# Feature Scaling
from sklearn.preprocessing import StandardScaler

sc_X = StandardScaler()
X_train = sc_X.fit_transform(X_train)
X_test = sc_X.transform(X_test)
sc_y = StandardScaler()
y_train = sc_y.fit_transform(y_train)
```

In [14]:
```
from sklearn.linear_model import LinearRegression
regressor = LinearRegression()
regressor.fit(X_train, y_train)
```

Out[14]:
```
LinearRegression()
```

In [15]:
```
# Predicting the Test set results
y_pred = regressor.predict(X_test)
```

In [16]:
```
print('DATOS DEL MODELO REGRESIÓN LINEAL SIMPLE')
print()
print('Valor de la pendiente o coeficiente "a":')
print(regressor.coef_)
print('Valor de la intersección o coeficiente "b":')
print(regressor.intercept_)
print()
print('La ecuación del modelo es igual a:')
print('y = ', regressor.coef_, 'x ', regressor.intercept_)
```

```
DATOS DEL MODELO REGRESIÓN LINEAL SIMPLE

Valor de la pendiente o coeficiente "a":
[[0.96860209]]
Valor de la intersección o coeficiente "b":
[-1.97125459e-16]

La ecuación del modelo es igual a:
y =  [[0.96860209]] x  [-1.97125459e-16]
```

In [17]:
```
print("Precisión del modelo:")
print(regressor.score(X_train, y_train))
```

```
Precisión del modelo:
0.9381900012894278
```

Creado por:

Isabel Maniega