POLYNOMIAL REGRESSION :

PAGE 126

FORMULA :

In [1]:
```
# y = b0+b1X1+b2X1(square)+b3X1(cube)+......+bnX1(power of 'n')
```

In [2]:
```
#
```

NOTE : POLYNOMIAL REGRESSION :

PAGE 128

In [3]:
```
#
```

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

In [12]:
```
dataset = pd.read_csv('DataS/Position_Salaries.csv')
X = dataset.iloc[:,1:-1].values
y = dataset.iloc[:,-1].values
```

In [13]:
```
X
```

Out[13]:
```
array([[ 1],
       [ 2],
       [ 3],
       [ 4],
       [ 5],
       [ 6],
       [ 7],
       [ 8],
       [ 9],
       [10]], dtype=int64)
```

In [14]:
```
y
```

Out[14]:
```
array([  45000,   50000,   60000,   80000,  110000,  150000,  200000,
        300000,  500000, 1000000], dtype=int64)
```

In [15]:
```
from sklearn.preprocessing import PolynomialFeatures
```

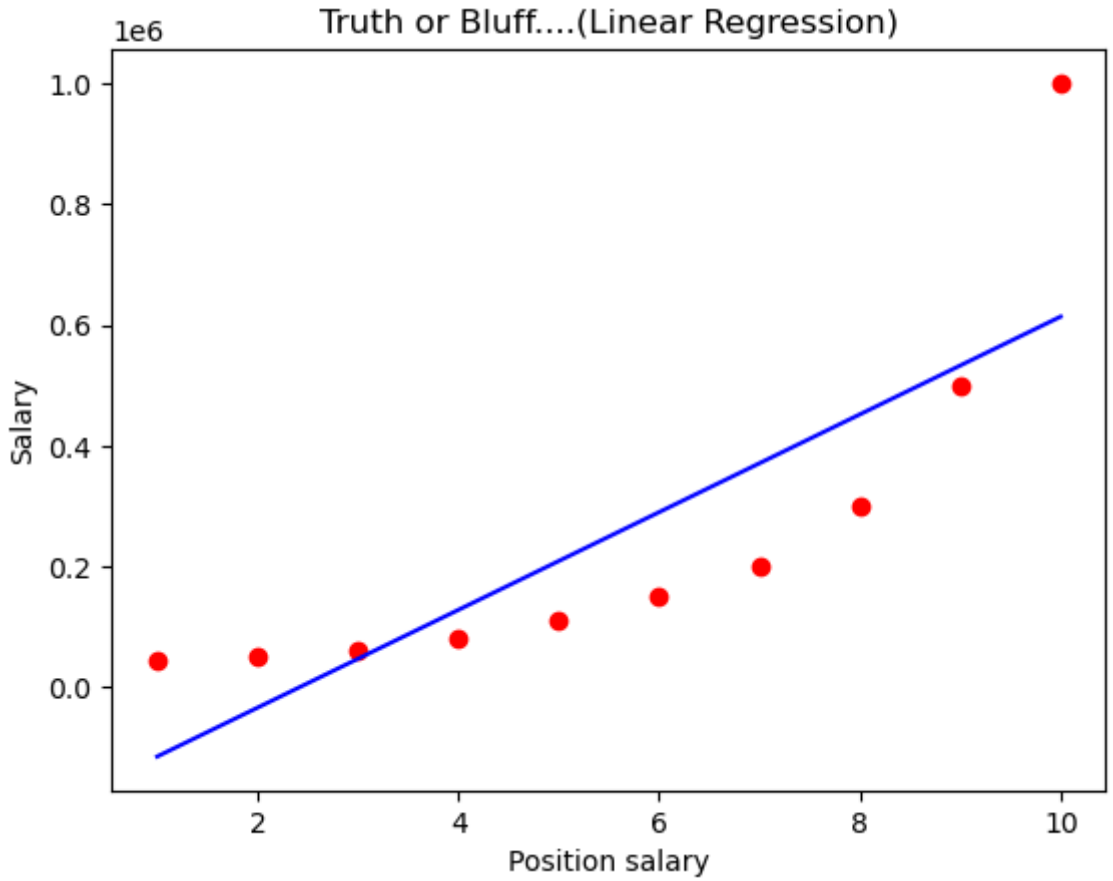
In [16]:
```
from sklearn.linear_model import LinearRegression
lin_reg = LinearRegression()
lin_reg.fit(X,y)
```

Out[16]:
```
LinearRegression()
```

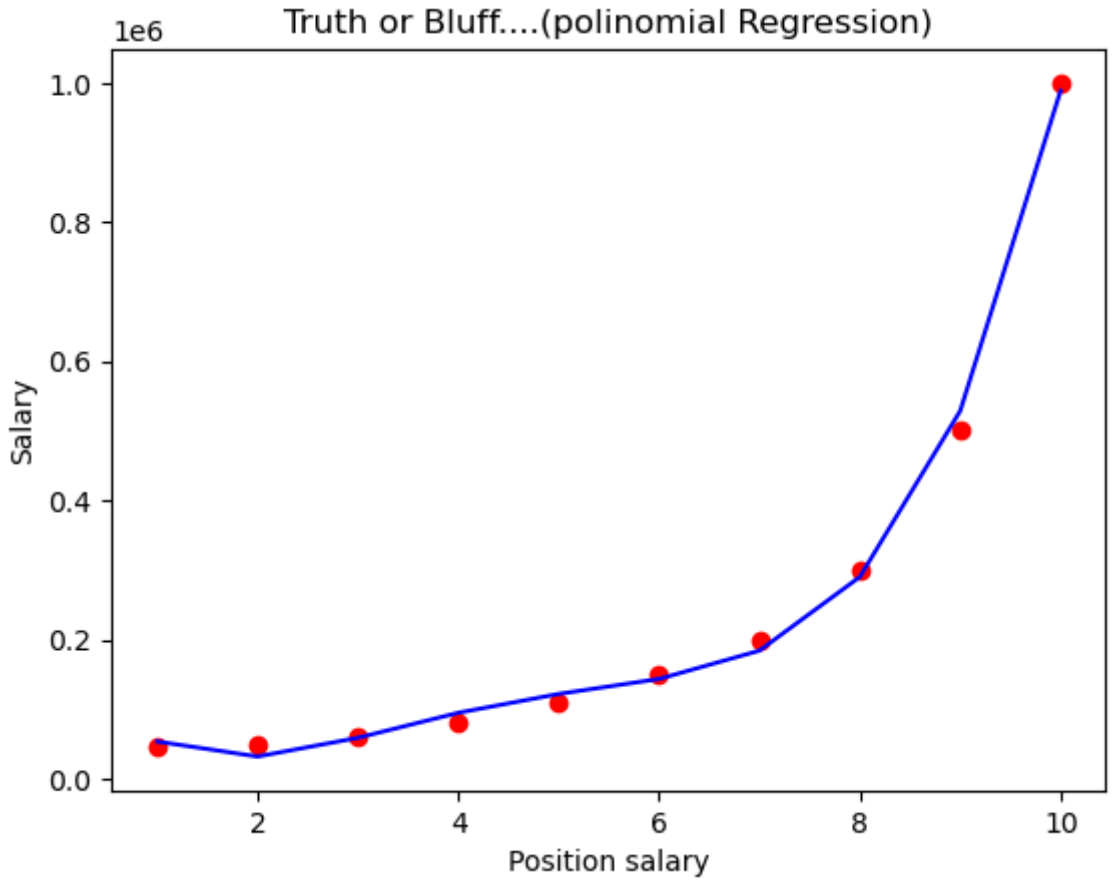
In [17]:
```
from sklearn.preprocessing import PolynomialFeatures
poly_reg =PolynomialFeatures(degree = 4)
x_poly = poly_reg.fit_transform(X)
lin_reg2= LinearRegression()
lin_reg2.fit(x_poly,y)
```

Out[17]:
```
LinearRegression()
```

In [18]:
```
plt.scatter(X,y, color = 'red')
plt.plot(X,lin_reg.predict(X),color = 'blue')
plt.title("Truth or Bluff....(Linear Regression)")
plt.xlabel("Position salary")
plt.ylabel("Salary")
plt.show()
```



In [19]:
```
plt.scatter(X,y, color = 'red')
plt.plot(X,lin_reg2.predict(poly_reg.fit_transform(X)),color = 'blue')
plt.title("Truth or Bluff....(polinomial Regression)")
plt.xlabel("Position salary")
plt.ylabel("Salary")
plt.show()
```



In [20]:
```
lin_reg.predict([[2.5]])
```

Out[20]:
```
array([6863.63636364])
```

In [21]:
```
lin_reg2.predict(poly_reg.fit_transform([[2.5]]))
```

Out[21]:
```
array([42102.54589164])
```

In [25]:
```
# Means, Here the position of '6.5' :

lin_reg.predict([[6.5]])
```

Out[25]:
```
array([330378.78787879])
```

In [26]:
```
lin_reg2.predict(poly_reg.fit_transform([[6.5]]))
```

Out[26]:
```
array([158862.45265153])
```

In [ ]:
```

```