In [2]:

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd

from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import PolynomialFeatures
```

In [4]:

```
dataset = pd.read_csv(r"C:\Users\AR ANSARI\NIT\ML\spyder\ML regression\emp_sal.c
dataset
```

Out[4]:

	Position	Level	Salary
0	Jr Software Engineer	1	45000
1	Sr Software Engineer	2	50000
2	Team Lead	3	60000
3	Manager	4	80000
4	Sr manager	5	110000
5	Region Manager	6	150000
6	AVP	7	200000
7	VP	8	300000
8	CTO	9	500000
9	CEO	10	1000000

In [5]:

```
X = dataset.iloc[:,1:2].values
X
```

Out[5]:

```
array([[ 1],
       [ 2],
       [ 3],
       [ 4],
       [ 5],
       [ 6],
       [ 7],
       [ 8],
       [ 9],
       [10]])
```

In [6]:

```
Y = dataset.iloc[:,2].values
Y
```

Out[6]:

```
array([  45000,   50000,   60000,   80000,  110000,  150000,  200000,
        300000,  500000, 1000000])
```

In [7]:

```
#Linear Regreassion
lin_reg = LinearRegression()
lin_reg.fit(X,Y)
```

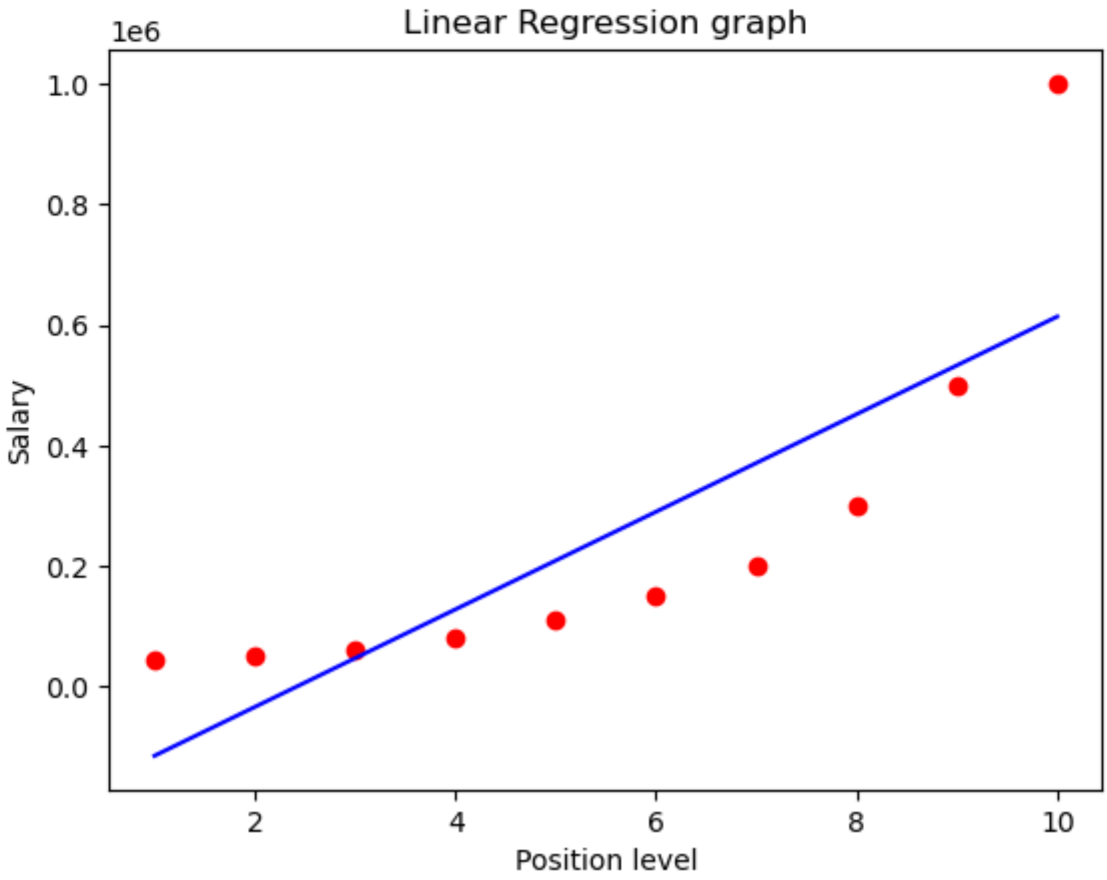
Out[7]:

```
▾ LinearRegression ⓘ

LinearRegression()
```

In [8]:

```
plt.scatter(X, Y, color='red')
plt.plot(X, lin_reg.predict(X), color='blue')
plt.title('Linear Regression graph')
plt.xlabel('Position level')
plt.ylabel('Salary')
plt.show()
```



In [10]:

```
poly_reg = PolynomialFeatures(degree = 4)  # You can try degree= 2 or 3 also
X_poly = poly_reg.fit_transform(X)
plt.show()
```

In [12]:

```
# again liner model build with 2nd degree
lin_reg_2 = LinearRegression()
lin_reg_2.fit(X_poly, Y)
```

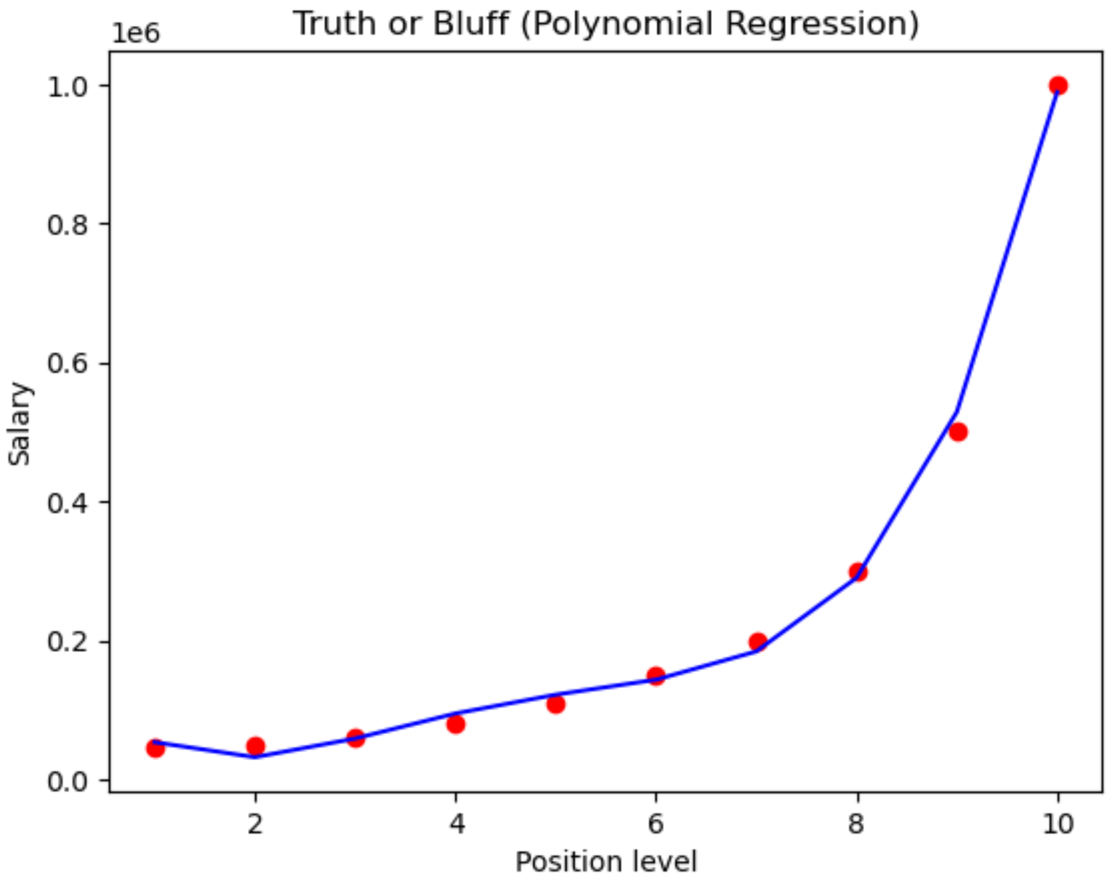
Out[12]:

```
▾ LinearRegression ⓘ

LinearRegression()
```

In [13]:

```
# Plot Polynomial Regressio (poly model)
plt.scatter(X, Y, color='red')
plt.plot(X, lin_reg_2.predict(poly_reg.fit_transform(X)), color='blue')
plt.title('Truth or Bluff (Polynomial Regression)')
plt.xlabel('Position level')   # Fixed: was plt.Xlabel
plt.ylabel('Salary')           # Fixed: was plt.Ylabel
plt.show()
```



In [14]:

```
 # Polynmial Model algrithm
lin_model_pred = lin_reg.predict([[6.5]])
print("Linear Model Prediction:", lin_model_pred)
```

```
Linear Model Prediction: [330378.78787879]
```

In [15]:

```
poly_model_pred = lin_reg_2.predict(poly_reg.fit_transform([[6.5]]))
print("Polynomial Model Prediction:", poly_model_pred)
```

```
Polynomial Model Prediction: [158862.4526516]
```

In [16]:

```

# ##### SVR Model Algorithm
from sklearn.svm import SVR
svr_model=SVR()
svr_model.fit(X,Y)
```

Out[16]:

```
▾ SVR ⓘ
SVR()
```

In [17]:

```
svr_model_pred=svr_model.predict([[6.5]])
print(svr_model_pred)
```

```
[130001.82883924]
```

In [18]:

```
#### KNN Model Algorithm

from sklearn.neighbors import KNeighborsRegressor
knn_model = KNeighborsRegressor(n_neighbors=4,weights='distance',algorithm='brut
knn_model.fit(X, Y)
```

Out[18]:

▾ KNeighborsRegressor

```
KNeighborsRegressor(algorithm='brute', n_neighbors=4, p=1,
weights='distance')
```

In [19]:

```
knn_model_pred = knn_model.predict([[6.5]])
print(knn_model_pred)
```

```
[182500.]
```

In [20]:

```
#### decission tree model agrithm
from sklearn.tree import DecisionTreeRegressor
dt_model = DecisionTreeRegressor()
dt_model.fit(X,Y)
```

Out[20]:

▾ DecisionTreeRegressor

```
DecisionTreeRegressor()
```

In [21]:

```
dt_model_pred = dt_model.predict([[6.5]])
print(dt_model_pred)
```

```
[150000.]
```

In [22]:

```
# Random Forest Algrithm

from sklearn.ensemble import RandomForestRegressor
rf_model = RandomForestRegressor()
rf_model.fit(X,Y)
```

Out[22]:

▾ RandomForestRegressor

```
RandomForestRegressor()
```

In [23]:

```
rf_model_pred = rf_model.predict([[6.5]])
print(rf_model_pred)
```

```
[162400.]
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

```
In [ ]:
```

```
In [ ]:
```

```
In [ ]:
```

```
In [ ]:
```

```
In [ ]:
```

```
In [ ]:
```