In [2]:
```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
```

In [3]:
```
dataset=pd.read_csv(r"E:\Naresh IT\data science class notes\machine learning\datase
```

In [4]:
```
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]:
```
#split the data into independent and dependent

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

In [6]:
```
dataset.columns
```

Out[6]:
```
Index(['Position', 'Level', 'Salary'], dtype='object')
```

In [5]:
```
x
```

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

In [6]:
```
y
```

Out[6]:
```
array([  45000,   50000,   60000,   80000,  110000,  150000,  200000,
        300000,  500000, 1000000])
```

In [7]:
```
from sklearn.linear_model import LinearRegression
lin_reg=LinearRegression()
lin_reg.fit(x,y)
```

Out[7]:
```
▾ LinearRegression
▶ Parameters
```

In [8]:
```
from sklearn.preprocessing import PolynomialFeatures
poly_reg=PolynomialFeatures(degree=4)
x_poly=poly_reg.fit_transform(x)
poly_reg.fit(x_poly,y)
```

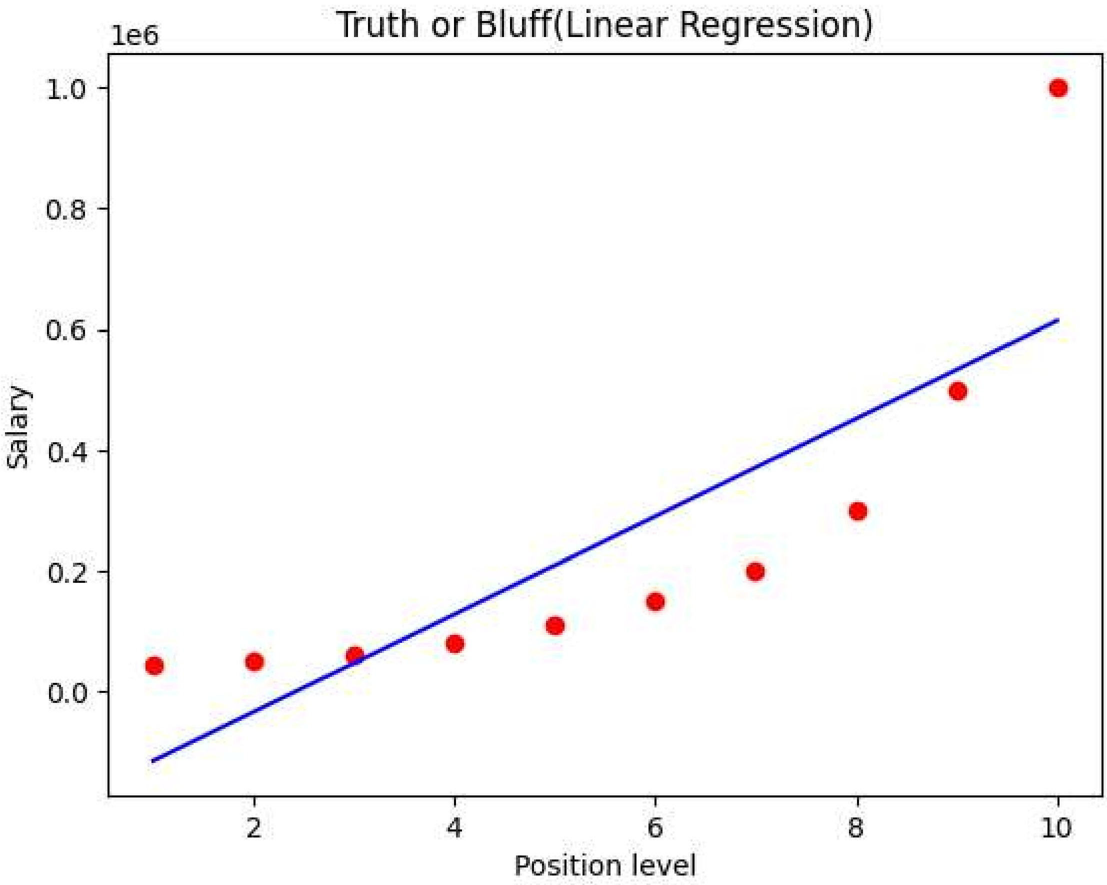
Out[8]:
```
▾ PolynomialFeatures
▶ Parameters
```

In [9]:
```
lin_reg_2=LinearRegression()
lin_reg_2.fit(x_poly,y)
```

Out[9]:
```
▾ LinearRegression
▶ Parameters
```

visualising the linear regression

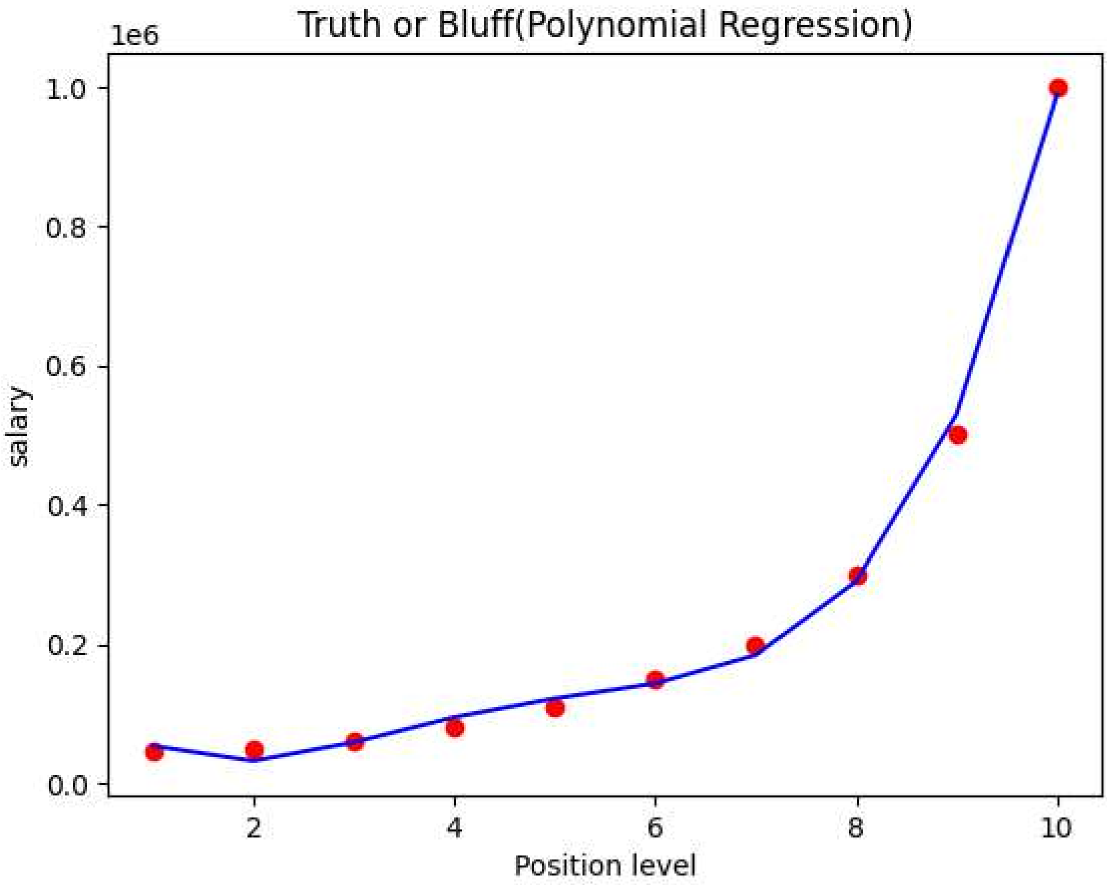
In [10]:
```
plt.scatter(x,y,color='red')
plt.plot(x,lin_reg.predict(x),color='blue')
plt.title("Truth or Bluff(Linear Regression)")
plt.xlabel("Position level")
plt.ylabel("Salary")
plt.show()
```



visualising the polynomial regression

In [13]:

```
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")
plt.ylabel("salary")
plt.show()
```



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

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

```
In [ ]: lin_reg_2.predict(poly_reg.fit_transform([[6.5]]))
```

```
Out[ ]: array([158862.45265155])
```

```
In [ ]: lin_model_pred = lin_reg.predict([[6.5]])
        lin_model_pred

        poly_model_pred = lin_reg_2.predict(poly_reg.fit_transform([[6.5]]))
        poly_model_pred
```

```
Out[ ]: array([158862.45265155])
```

```
In [ ]: from sklearn.svm import SVR
        svr_model=SVR(kernel='poly',degree=4,gamma='auto',C=10.0)
        svr_model.fit(x,y)

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

```
[175705.60452113]
```

```
In [ ]: from sklearn.neighbors import KNeighborsRegressor
        knn_reg_model=KNeighborsRegressor(n_neighbors=5,weights='distance',p=2)
        knn_reg_model.fit(x,y)
```

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

```
[175348.8372093]
```

In [ ]:

```
from sklearn.tree import DecisionTreeRegressor
dt_model=DecisionTreeRegressor()
dt_model.fit(x,y)

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

```
[150000.]
```

In [ ]:

```
from sklearn.ensemble import RandomForestRegressor
rf_model=RandomForestRegressor(n_estimators=26,random_state=0)
rf_model.fit(x,y)

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

```
[166538.46153846]
```