

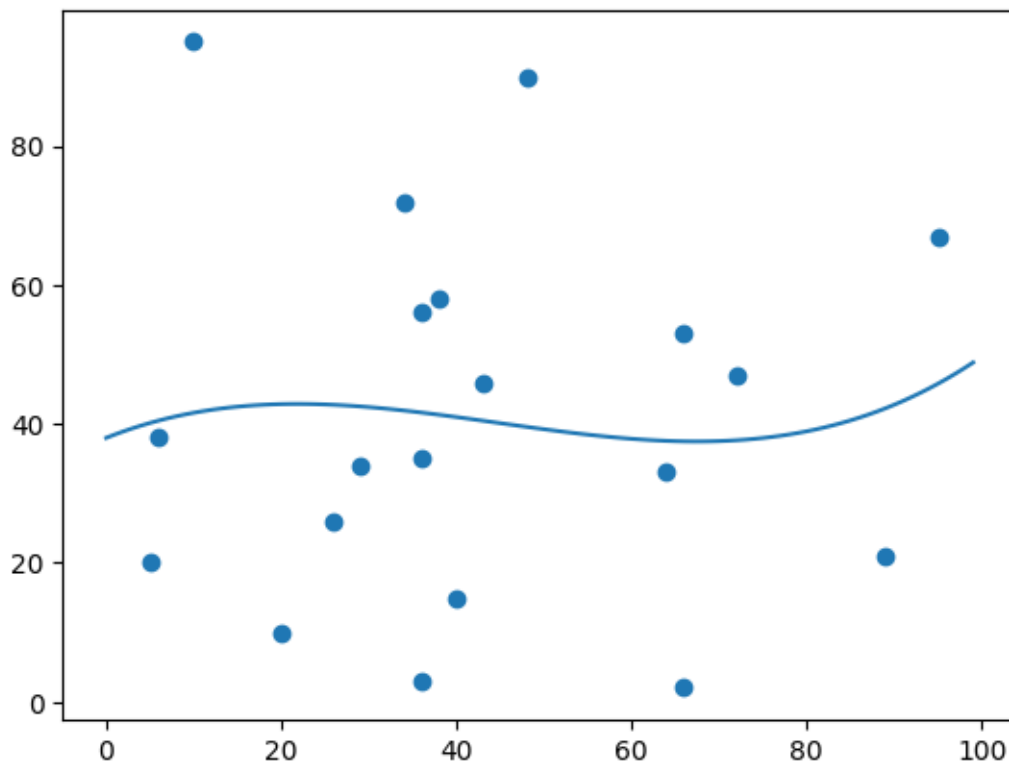
untitled8

June 20, 2023

POLYNOMIAL REGRESSION

1- BAD FIT

```
[2]: import numpy as np
import matplotlib.pyplot as plt
x=[89,43,36,36,95,10,66,34,38,20,26,29,48,64,6,5,36,66,72,40]
y=[21,46,3,35,67,95,53,72,58,10,26,34,90,33,38,20,56,2,47,15]
model = np.poly1d(np.polyfit(x,y,3)) # 3 degree curve
myline = np.linspace(1,95,100) # 100 is showing no of sample point
plt.scatter(x,y)
plt.plot(model(myline))
plt.show()
```

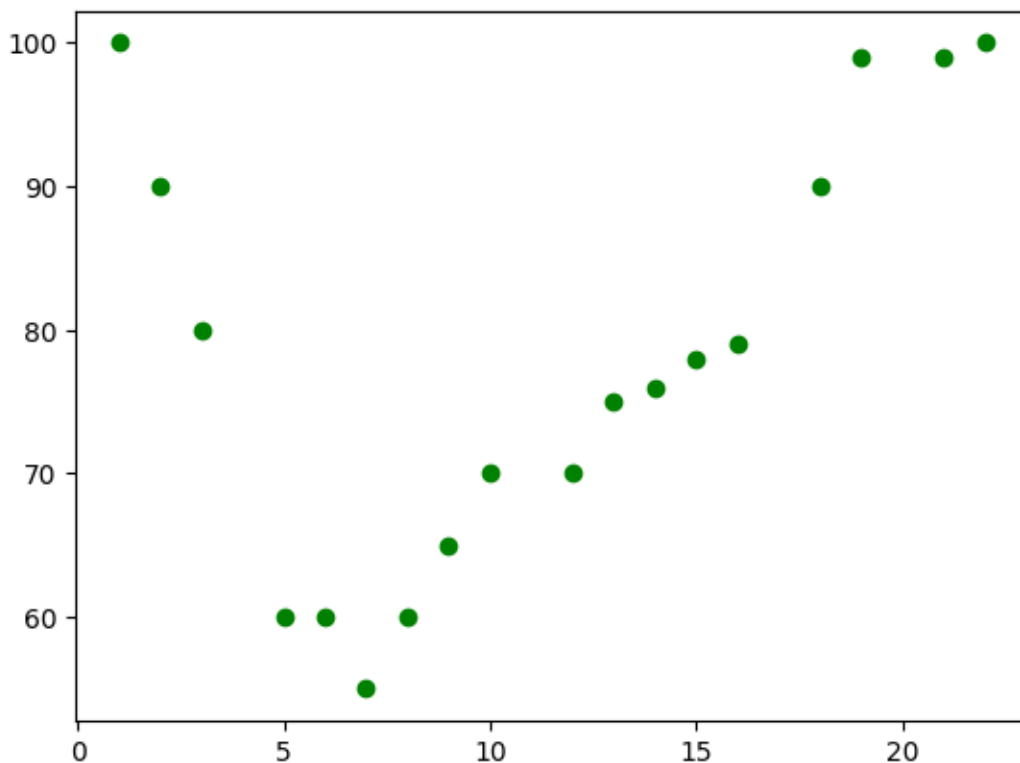


```
[4]: # R square value
from sklearn.metrics import r2_score
print(r2_score(y,model(x)))
```

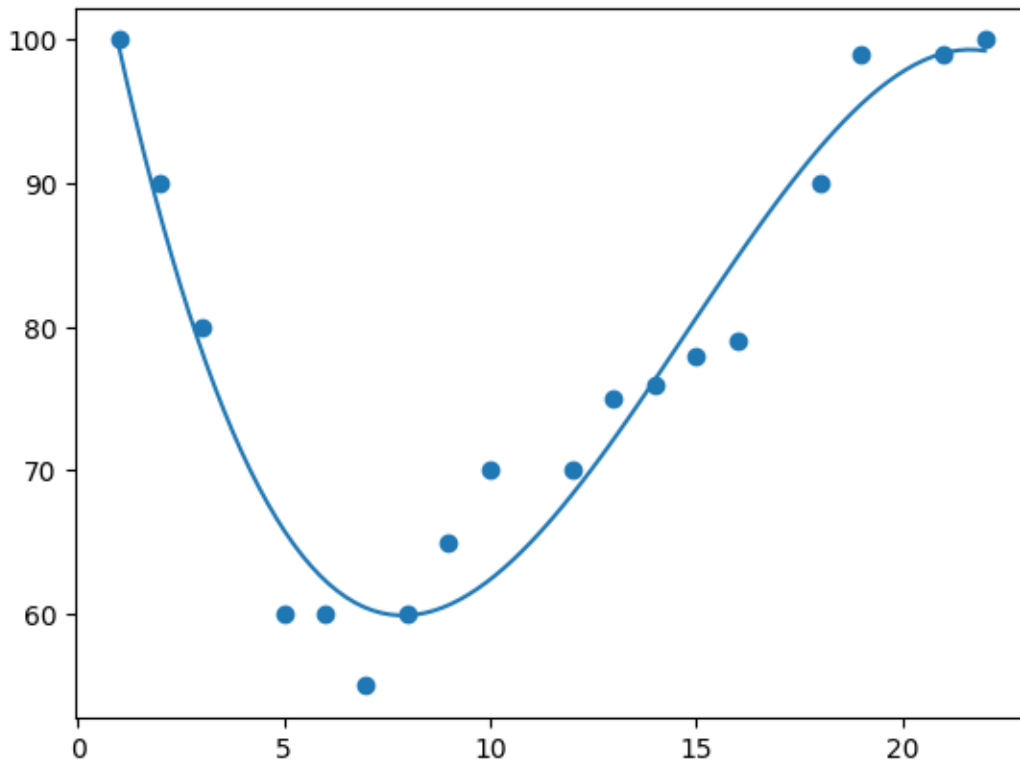
0.009952707566680652

BEST FIT

```
[5]: # Step-1 Data
import matplotlib.pyplot as plt
x = [1,2,3,5,6,7,8,9,10,12,13,14,15,16,18,19,21,22]
y = [100,90,80,60,60,55,60,65,70,70,75,76,78,79,90,99,99,100]
plt.scatter(x,y, color ="green")
plt.show()
```



```
[6]: # Step-2 Darw line
model = np.poly1d(np.polyfit(x,y,3)) # 3 degree curve
myline = np.linspace(1,22,100) # 100 is no of sample points showing
plt.scatter(x,y)
plt.plot(myline, model(myline))
plt.show()
```



```
[7]: # step-3 Required
from sklearn.metrics import r2_score
print(r2_score(y,model(x)))
```

0.9432150416451026

```
[8]: # Prediction
model = np.poly1d(np.polyfit(x,y,3))
pred = model(1)
print(pred)
```

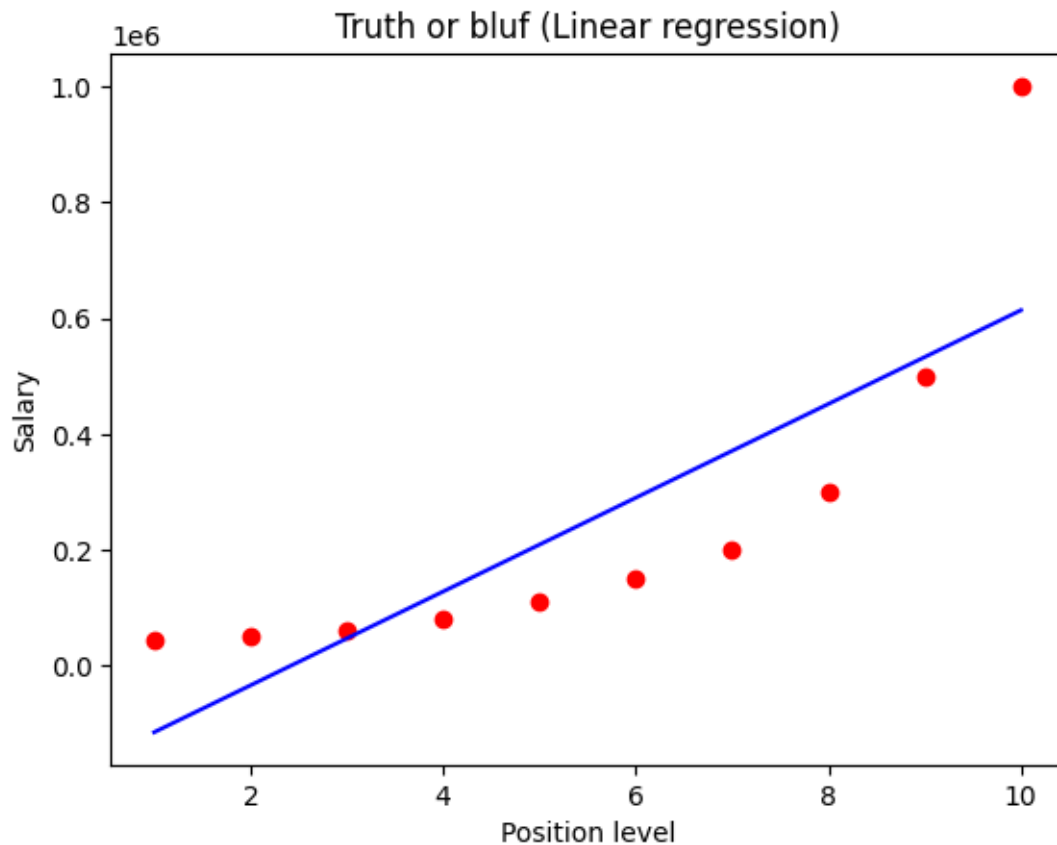
99.54274392967326

3- Hands on Example

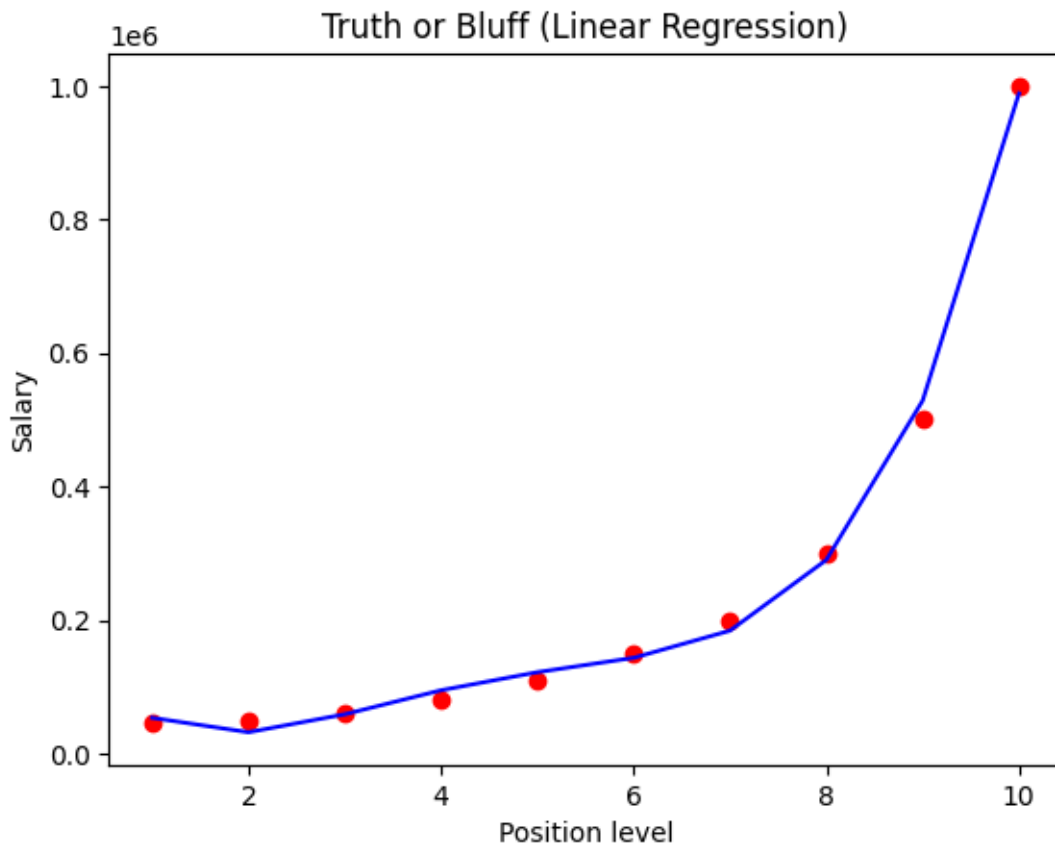
```
[16]: # Another important example
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
dataset = pd.read_csv('https://s3.us-west-2.amazonaws.com/public.gamelab.fun/
↳dataset/position_salaries.csv')
X= dataset[['Level']]
y= dataset['Salary']
```

```
[20]: # splitting data set into training and testing
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,y,test_size=0.
↪2,random_state=0)
```

```
[26]: # fitting linear regression to dataset
from sklearn.linear_model import LinearRegression
lin_reg = LinearRegression().fit(X,y)
# Visualizing the linear regression model result
def viz_linear():
    plt.scatter(X,y,color="red")
    plt.plot(X,lin_reg.predict(X),color="blue")
    plt.title("Truth or bluf (Linear regression)")
    plt.xlabel("Position level")
    plt.ylabel("Salary")
    plt.show()
    return
viz_linear()
```



```
[28]: # Fitting Polynomial Regression to the dataset
from sklearn.preprocessing import PolynomialFeatures
poly_reg = PolynomialFeatures(degree=4)
X_poly = poly_reg.fit_transform(X)
pol_reg = LinearRegression()
pol_reg.fit(X_poly, y)
# Visualizing the Polyomial Regression results
def viz_polymonial():
    plt.scatter(X, y, color='red')
    plt.plot(X, pol_reg.predict(poly_reg.fit_transform(X)), color='blue')
    plt.title('Truth or Bluff (Linear Regression)')
    plt.xlabel('Position level')
    plt.ylabel('Salary')
    plt.show()
    return
viz_polymonial()
```



```
[ ]:
```