```
#write machine learning algorithm to predict salary of an employee(no linearly distributed)using polynomial regression
```

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

```
from google.colab import files
uploaded = files.upload()
```

Choose files No file chosen Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

```
Saving 50_Startups.csv to 50_Startups.csv
```

```
from google.colab import files
uploaded = files.upload()
```

Choose files No file chosen Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

```
Saving Position_Salaries.csv to Position_Salaries.csv
```

```
dataset = pd.read_csv('Position_Salaries.csv')
```

Show hidden output

```
dataset.head()
```

	Position	Level	Salary
0	Business Analyst	1	45000
1	Junior Consultant	2	50000
2	Senior Consultant	3	60000
3	Manager	4	80000
4	Country Manager	5	110000

```
dataset =dataset.drop(['Position'], axis=1)
```

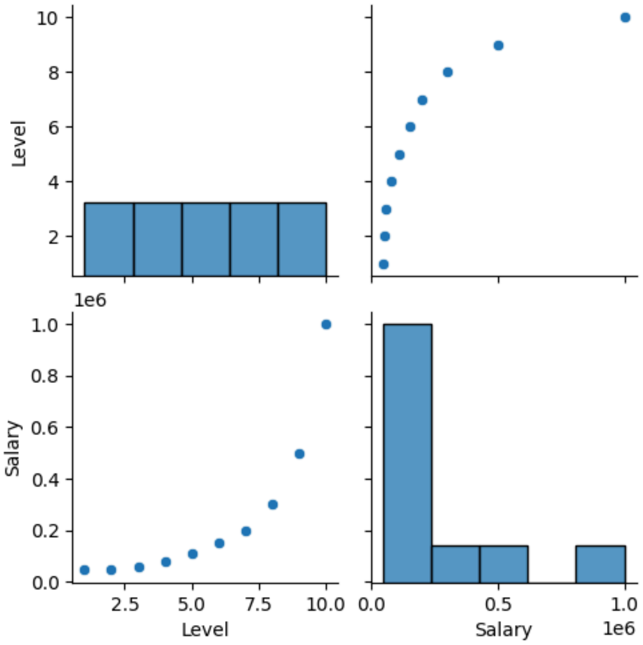
```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
/tmp/ipython-input-2311325117.py in <cell line: 0>()
----> 1 dataset =dataset.drop(['Position'], axis=1)

                                  3 frames
/usr/local/lib/python3.12/dist-packages/pandas/core/indexes/base.py in drop(self, labels, errors)
   7068         if mask.any():
   7069             if errors != "ignore":
-> 7070                 raise KeyError(f"{labels[mask].tolist()} not found in axis")
   7071             indexer = indexer[~mask]
   7072         return self.delete(indexer)

KeyError: "['Position'] not found in axis"
```

```
sns.pairplot(dataset)
```

```
<seaborn.axisgrid.PairGrid at 0x7faab50aaae0>
```



```
x= dataset.drop(['Salary'], axis=True)
y= dataset['Salary']
```

```
#now split_The Data
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=42)
```

```
x_train.shape,x_test.shape,y_train.shape,y_test.shape
```

```
((8, 1), (2, 1), (8,), (2,))
```

```
from sklearn.linear_model import LinearRegression
lr = LinearRegression()
lr.fit(x_train,y_train)
```

```
▾ LinearRegression ⓘ ?
LinearRegression()
```

```
print("Training Accuracy :", lr.score(x_train, y_train))
print("Testing Accuracy:", lr.score(x_test, y_test))
```

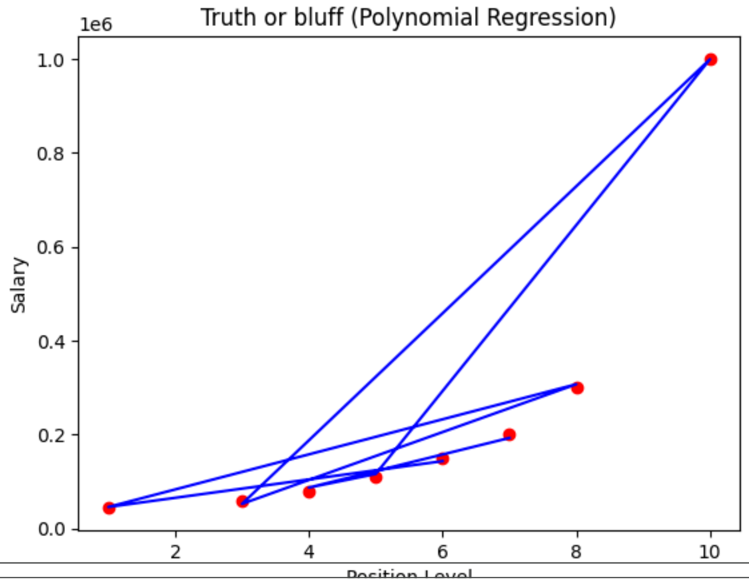
```
Training Accuracy : 0.6366049276570868
Testing Accuracy: 0.8451346684575974
```

```
from scipy.linalg import polar
#Training the polynomial
from sklearn.preprocessing import PolynomialFeatures
poly = PolynomialFeatures(degree=4)
x_poly = poly.fit_transform(x_train)
lin_reg_2 = LinearRegression()
lin_reg_2.fit(x_poly, y_train)
x_polytest = poly.fit_transform(x_test)
```

```
print("Polynomial Regression Training Accuracy :", lin_reg_2.score(x_poly, y_train))
print("Polynomial Regression Testing Accuracy:", lin_reg_2.score(poly.fit_transform(x_test), y_test))
```

```
Polynomial Regression Training Accuracy : 0.9995857211026754
Polynomial Regression Testing Accuracy: 0.9714666803842444
```

```
plt.scatter(x_train, y_train, color='red')
plt.plot(x_train, lin_reg_2.predict(poly.fit_transform(x_train)), color='blue') # Plotting Polynomial Regression
plt.title("Truth or bluff (Polynomial Regression)")
plt.xlabel('Position Level')
plt.ylabel('Salary')
plt.show()
```

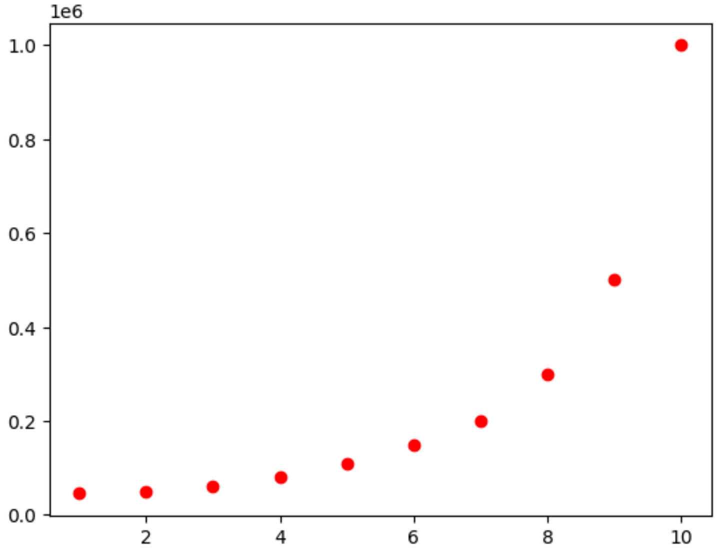


```
# Create a smooth curve for the polynomial regression line


plt.scatter(x, y, color='red')
plt.plot(X_grid, lin_reg_2.predict(poly.fit_transform(X_grid)), color='blue')
plt.title("Truth or Bluff (Polynomial Regression)")
plt.xlabel('Position Level')
plt.ylabel('Salary')
plt.show()
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
/tmp/ipython-input-3026209953.py in <cell line: 0>()
      3
      4 plt.scatter(x, y, color='red')
----> 5 plt.plot(X_grid, lin_reg_2.predict(poly.fit_transform(X_grid)), color='blue')
      6 plt.title("Truth or Bluff (Polynomial Regression)")
      7 plt.xlabel('Position Level')

NameError: name 'X_grid' is not defined
```


1e6
1.0
0.8
0.6
0.4
0.2
0.0
2
4
6
8
10