Regression

Python Programming Lab

05506231 Statistics and Probability

Asst. Prof. Dr.Anantaporn Hanskunatai

Outline

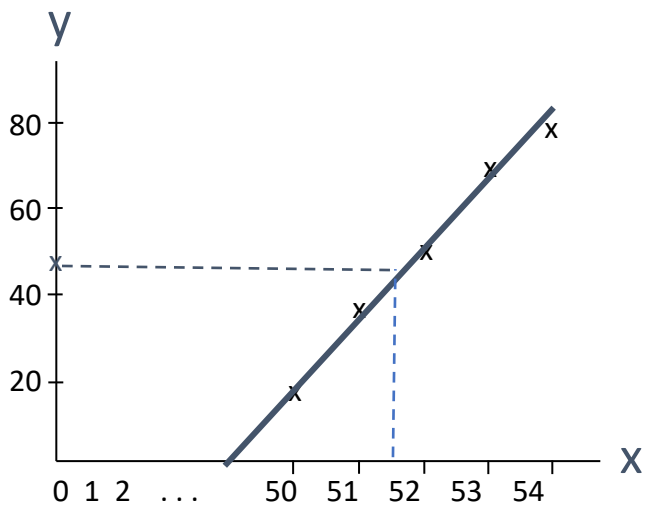
- Simple linear regression
- Case study on linear regression
- Polynomial regression
- Case study on polynomial regression

Simple Linear Regression

- Find linear regression of the following data (n = 5) (from slide #7)

y	x
20	50
40	51
50	52
70	53
80	54

$\hat{Y}_i = a + b\,x_i = -728 + 15x_i$



- Predict y-value (at x = 51.5) = -728 + 15(51.5) = 44.5

Simple Linear Regression

```
import pandas as pd
import numpy as np
from sklearn.linear_model import LinearRegression
```

```
x = np.array([50, 51, 52, 53, 54]).reshape(-1,1)
y = np.array([20, 40, 50, 70, 80]).reshape(-1,1)
```

y	x
20	50
40	51
50	52
70	53
80	54

```
model=LinearRegression()
model.fit(x,y)
```

```
print('intercept:', model.intercept_)
print('slope:', model.coef_)
```

```
intercept: [-728.]
slope: [[15.]]
```

$$\hat{Y} = -728 + 15x$$

Predict new input x

```
y_predict=model.predict([[51.5]])
y_predict
```

```
array([[44.5]])
```

Case study on Linear Regression

- advertising.csv

	A	B	C	D	E
1		TV	Radio	Newspaper	Sales
2	1	230.1	37.8	69.2	22.1
3	2	44.5	39.3	45.1	10.4
4	3	17.2	45.9	69.3	9.3
5	4	151.5	41.3	58.5	18.5
6	5	180.8	10.8	58.4	12.9
7	6	8.7	48.9	75	7.2
8	7	57.5	32.8	23.5	11.8
9	8	120.2	19.6	11.6	13.2
10	9	8.6	2.1	1	4.8
11	10	199.8	2.6	21.2	10.6
12	11	66.1	5.8	24.2	8.6
13	12	214.7	24	4	17.4
14	13	23.8	35.1	65.9	9.2
15	14	97.5	7.6	7.2	9.7
16	15	204.1	32.9	46	19
17	16	195.4	47.7	52.9	22.4

- 5 columns and 200 rows
- TV Radio and Newspaper columns represent the budget for advertising in these channels
- Sales column represents sales data

Case study on Linear Regression

- Use simple linear regression to predict the sales based on the amount which is company spend for TV advertising

```
import pandas as pd
from sklearn.linear_model import LinearRegression
```

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

```
df= pd.read_csv( 'advertising.csv' )
```

Data transformation

```
x_TV=df.TV.values.reshape(-1,1)
y=df.Sales.values.reshape(-1,1)
```

Case study on Linear Regression

Create simple linear regression model

```
model=LinearRegression()
model.fit(x_TV,y)
```

View the model

```
model.intercept_, model.coef_
```

```
model.intercept_, model.coef_

(array([7.03259355]), array([[0.04753664]]))
```

Prediction

```
newX=[[300],[500],[1000]]
y_predict=model.predict(newX)
```

```
y_predict

array([[21.29358568],
       [30.80091377],
       [54.56923398]])
```

$$\hat{Y} = 7.0326 + (0.0475 * TV)$$

Case study on Linear Regression

Model Evaluation

- Coefficient of determination (R^2)
  - R^2 = 1 means modeled values exactly match the observed values

$$R^2 = 1 - \frac{RSS}{TSS}$$

R^2 = coefficient of determination
RSS = sum of squares of residuals
TSS = total sum of squares

$$RSS = \sum_{i=1}^{n}(y_i - f(x_i))^2$$

RSS = residual sum of squares
y_i = i^th value of the variable to be predicted
$f(x_i)$ = predicted value of y_i
n = upper limit of summation

$$\text{TSS} = \sum_{i=1}^{n}(y_i - \bar{y})^2$$

TSS = total sum of squares
n = number of observations
y_i = value in a sample
$\bar{y}$ = mean value of a sample

```
model.score(x_TV,y)
```

```
0.611875050850071
```

Case study on Linear Regression

Model Evaluation

- Mean Absolute Error (MAE)
- Mean Square Error (MSE)

$$\text{MAE} = \frac{\sum_{i=1}^{n} |y_i - x_i|}{n}$$

MAE = mean absolute error
y_i = prediction
x_i = true value
n = total number of data points

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

MSE = mean squared error
n = number of data points
Y_i = observed values
$\hat{Y}_i$ = predicted values

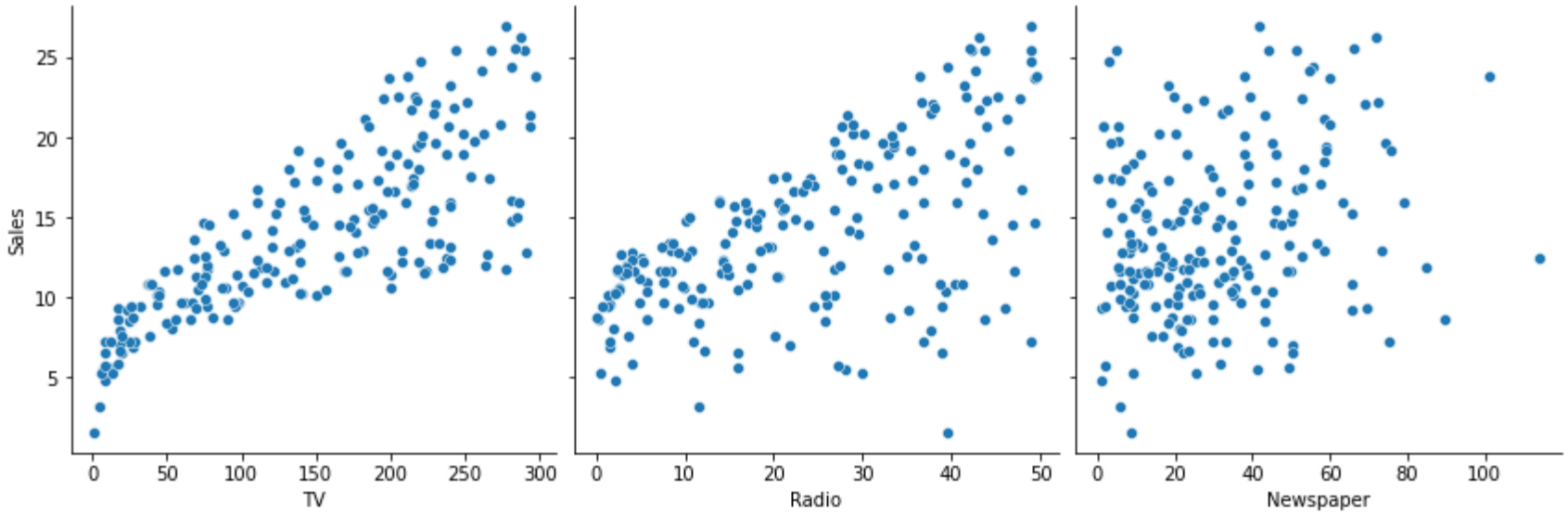
```
from sklearn.metrics import mean_squared_error, mean_absolute_error
y_predict1=model.predict(x_TV)
print('MAE =', mean_absolute_error(y,y_predict1))
print('MSE =', mean_squared_error(y,y_predict1))
```

```
MAE = 2.549806038927486
MSE = 10.512652915656757
```

Case study on Linear Regression

- Use multiple linear regression to predict the sales based on the amount which is company spend for each channels
- Data visualization

```
import seaborn as sns
sns.pairplot(df, x_vars=['TV','Radio','Newspaper'], y_vars='Sales', height=4)
```



Case study on Linear Regression

- Forward Selection
  - Step 1

```
x_Radio=df.Radio.values.reshape(-1,1)
x_News=df.Newspaper.values.reshape(-1,1)
```

```
model.fit(x_Radio,y)
print(model.score(x_Radio,y))
```

```
model.fit(x_News,y)
print(model.score(x_News,y))
```

Input	R2
TV	0.611875
Radio	0.332032
Newspaper	0.052120

Case study on Linear Regression

- Forward Selection
  - Step 2

```
x_TVRadio=df[['TV','Radio']]
x_TVNews=df[['TV','Newspaper']]
```

```
model.fit(x_TVRadio,y)
print(model.score(x_TVRadio,y))
```

```
model.fit(x_TVNews,y)
print(model.score(x_TVNews,y))
```

  - Step 3

```
X3=df[['TV','Radio','Newspaper']]
model.fit(X3,y)
print(model.score(X3,y))
```

Input	R2
TV	0.611875
Radio	0.332032
Newspaper	0.052120
TV, Radio	0.897194
TV, Newspaper	0.6458355
TV, Radio, Newspaper	0.8972106

Case study on Linear Regression

- Multiple linear regression

```
print(model.coef_)
print(model.intercept_)
```

```
[[ 0.04576465  0.18853002 -0.00103749]]
[2.93888937]
```

$$\hat{Y} = 2.9389 + (0.0458 * TV) + (0.1885 * \text{Radio}) + (-0.001 * \text{Newspaper})$$

Case study on Linear Regression

- Prediction

No.	TV	Radio	Newspaper	Sales	Sales
Case 1	300	0	0	?	**16.67**
Case 2	0	300	0	?	**59.50**
Case 3	0	0	300	?	**2.63**
Case 4	100	200	200	?	**45.01**
Case 5	100	200	0	?	**45.22**

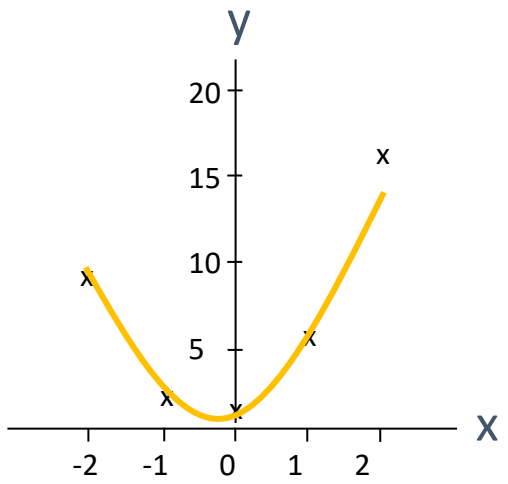
```
x_input=[[300,0,0],[0,300,0],[0,0,300],[100,200,200],[100,200,0]]
model.predict(x_input)
```

```
array([16.66828301, 59.49789444,  2.62764146, 45.01385869, 45.2213573 ])
```

Polynomial Regression

- Find Polynomial Regression of the following data (from slide #10)

y	x
1	0
6	1
17	2
2	-1
9	-2



$$\hat{Y} = a + bx + cx^2 = 1 + 2x + 3x^2$$

Polynomial Regression

```
import numpy as np
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import PolynomialFeatures
```

```
x = np.array([0, 1, 2, -1, -2]).reshape(-1,1)
y = np.array([1, 6, 17, 2, 9]).reshape(-1,1)
```

```
poly_features=PolynomialFeatures(degree=2)
x_poly=poly_features.fit_transform(x)
model=LinearRegression()
model.fit(x_poly,y)
```

```
print('intercept:', model.intercept_)
print('slope:', model.coef_)
```

```
intercept: [1.]
slope: [[0. 2. 3.]]
```

y	x
1	0
6	1
17	2
2	-1
9	-2

$$\hat{Y} = 1 + 2x + 3x^2$$

Case study on Polynomial Regression

- salary.csv

	A	B	C
1	Position	Level	Salary
2	Business Analyst	1	45,000.00
3	Junior Consultant	2	50,000.00
4	Senior Consultant	3	60,000.00
5	Manager	4	80,000.00
6	Country Manager	5	110,000.00
7	Region Manager	6	150,000.00
8	Partner	7	200,000.00
9	Senior Partner	8	300,000.00
10	C-level	9	500,000.00
11	CEO	10	1,000,000.00

Case study on Polynomial Regression

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import PolynomialFeatures
```

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

```
df= pd.read_csv('salary.csv')
```

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

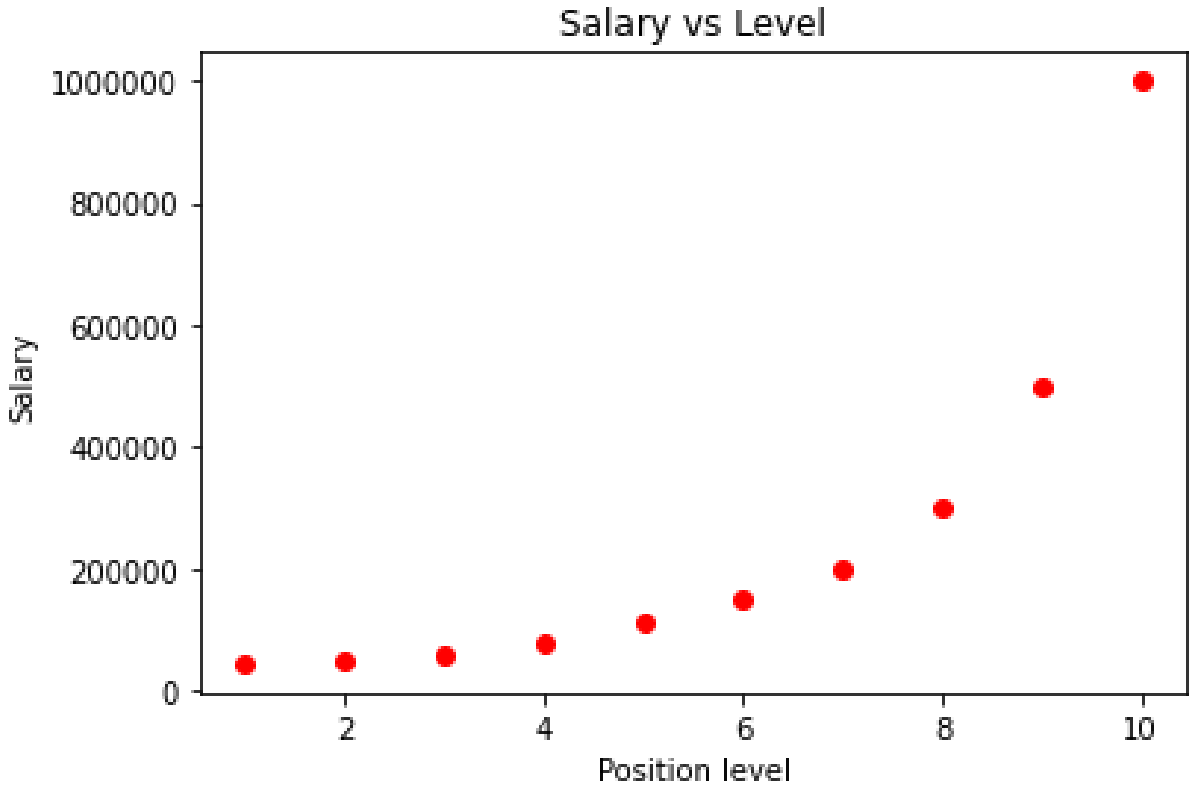
```
df
```

	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
5	Region Manager	6	150000
6	Partner	7	200000
7	Senior Partner	8	300000
8	C-level	9	500000
9	CEO	10	1000000

Case study on Polynomial Regression

- Data visualization

```
plt.scatter(x,y, color='red')
plt.ticklabel_format(style='plain')
plt.title('Salary vs Level')
plt.xlabel('Position level')
plt.ylabel('Salary')
plt.show()
```



Case study on Polynomial Regression

```
degree=['Degree1', 'Degree2', 'Degree3', 'Degree4']
Predict=pd.DataFrame(index=degree).T
Rscore = []
```

```
for k in range(1, 5):
  poly_features=PolynomialFeatures(degree=k)
  x_poly=poly_features.fit_transform(x)
  model=LinearRegression()
  model.fit(x_poly,y)
  p1=model.predict(x_poly)
  if(k==1):
    Predict.Degree1=p1
  elif(k==2):
    Predict.Degree2=p1
  elif(k==3):
    Predict.Degree3=p1
  else:
   Predict.Degree4=p1
  Rscore.append(model.score(x_poly,y))
```

Case study on Polynomial Regression

Predict

	Degree1	Degree2	Degree3	Degree4
0	-114454.545455	118727.272727	14902.097902	53356.643357
1	-33575.757576	44151.515152	78759.906760	31759.906760
2	47303.030303	8439.393939	94960.372960	58642.191142
3	128181.818182	11590.909091	88223.776224	94632.867133
4	209060.606061	53606.060606	83270.396270	121724.941725
5	289939.393939	134484.848485	104820.512821	143275.058275
6	370818.181818	254227.272727	177594.405594	184003.496504
7	451696.969697	412833.333333	326312.354312	289994.172494
8	532575.757576	610303.030303	575694.638695	528694.638695
9	613454.545455	846636.363636	950461.538462	988916.083916

Rscore

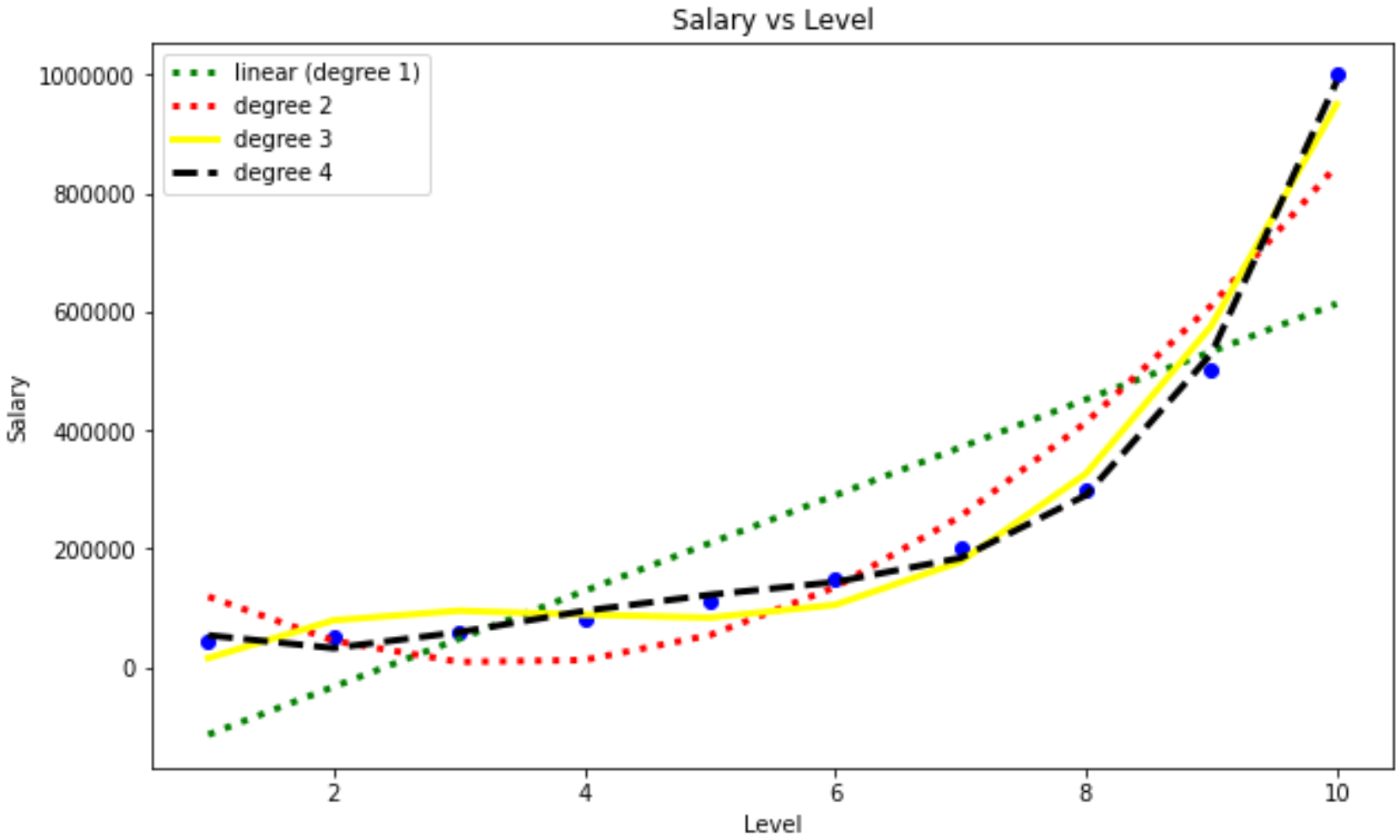
```
[0.6690412331929895,
 0.9162082221443942,
 0.9812097727913367,
 0.9973922891706614]
```

Case study on Polynomial Regression

- Visualization of model prediction

```
plt.figure(figsize=(10,6))
plt.ticklabel_format(style='plain')
plt.scatter(x,y,color='b')
plt.plot(x,Predict.Degree1,linewidth='3',color='g',linestyle='dotted',label='linear(degree 1)')
plt.plot(x,Predict.Degree2,linewidth='3',color='r',linestyle='dotted',label='degree 2')
plt.plot(x,Predict.Degree3,linewidth='3',color='yellow', linestyle='solid',label='degree 3')
plt.plot(x,Predict.Degree4,linewidth='3',color='black',linestyle='dashed',label='degree 4')
plt.legend(loc = 'best')
plt.title('Salary vs Level')
plt.xlabel('Level')
plt.ylabel('Salary')
plt.show()
```

Case study on Polynomial Regression



Case study on Polynomial Regression

- Model

```
print(model.intercept_, model.coef_)
```

```
print(model.intercept_, model.coef_)

184166.66666719693 [      0.         -211002.33100292   94765.44289063  -15463.28671331
     890.15151515]
```

- Prediction

```
x_poly = PolynomialFeatures(degree=4)
model.predict(x_poly.fit_transform([[6.5]]))
```

```
array([158862.45265155])
```