Lez03_Ridge

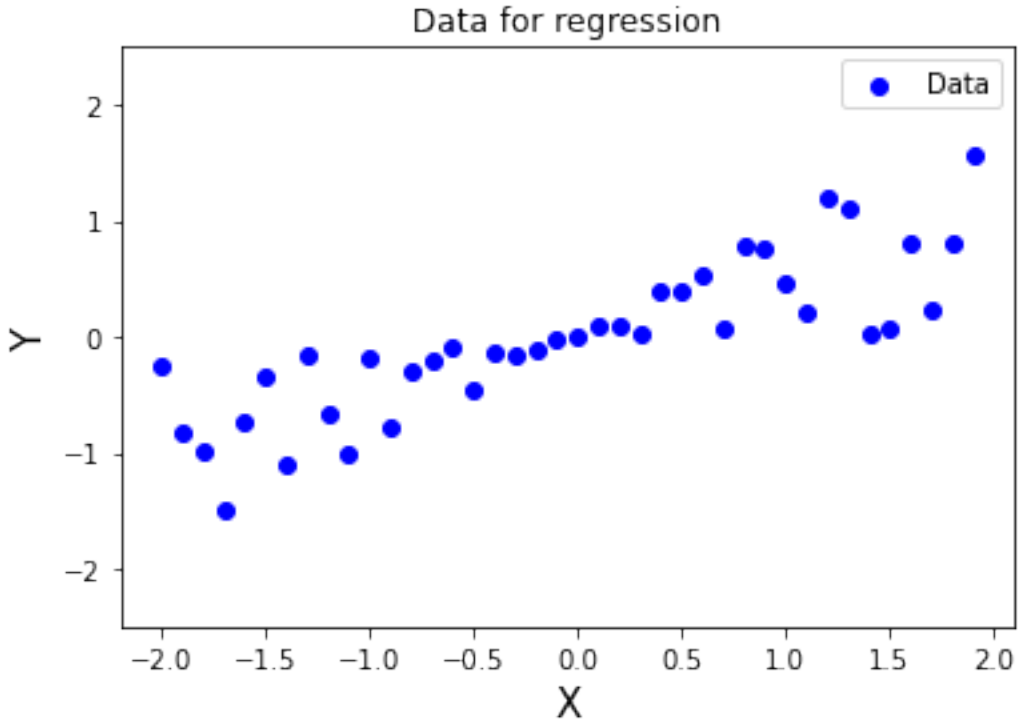
October 16, 2022

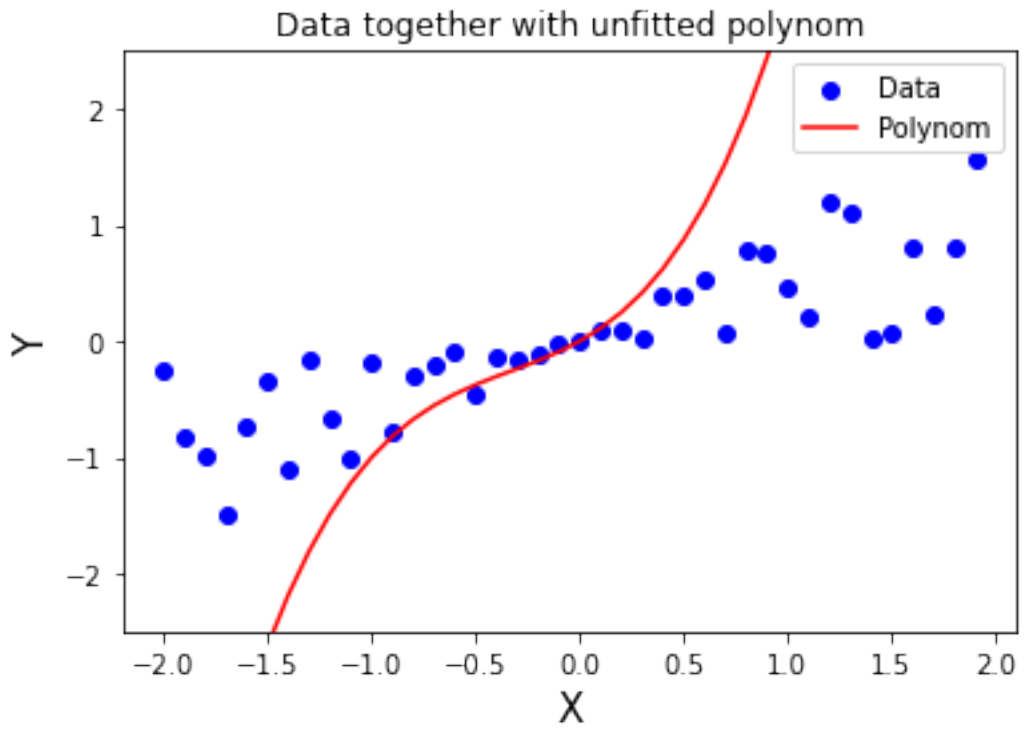
[107]:
```
#Import libraries
import numpy as np
import matplotlib.pyplot as plt
import random
from sklearn.linear_model import Ridge
from sklearn.preprocessing import PolynomialFeatures

#Variables
x = np.arange(-2.0, 2.0, 0.1)
y = np.arange(-2.0, 2.0, 0.1)#Initialise

for i in range(np.size(y)): #Random data for y
    y[i] *= random.random()

p = int(random.random()*5) #Random degree of polynom between 0 and 5
print('Degree of the polynom:', p)

#Plotting raw data
plt.scatter(x,y, c='b', label = "Data")

plt.xlabel("X", size = 15)
plt.ylabel("Y", size = 15)
plt.ylim((-2.5,2.5))
plt.title("Data for regression")
plt.legend()
plt.show()

#Create polynom
polynom = x * 0 #Initialise array

for i in range(p+1):
    if(i != 0):
        for j in range (np.size(polynom)):
            polynom[j] += x[j]**i

#Plotting data with polynom
plt.scatter(x,y, c='b', label = "Data")
```

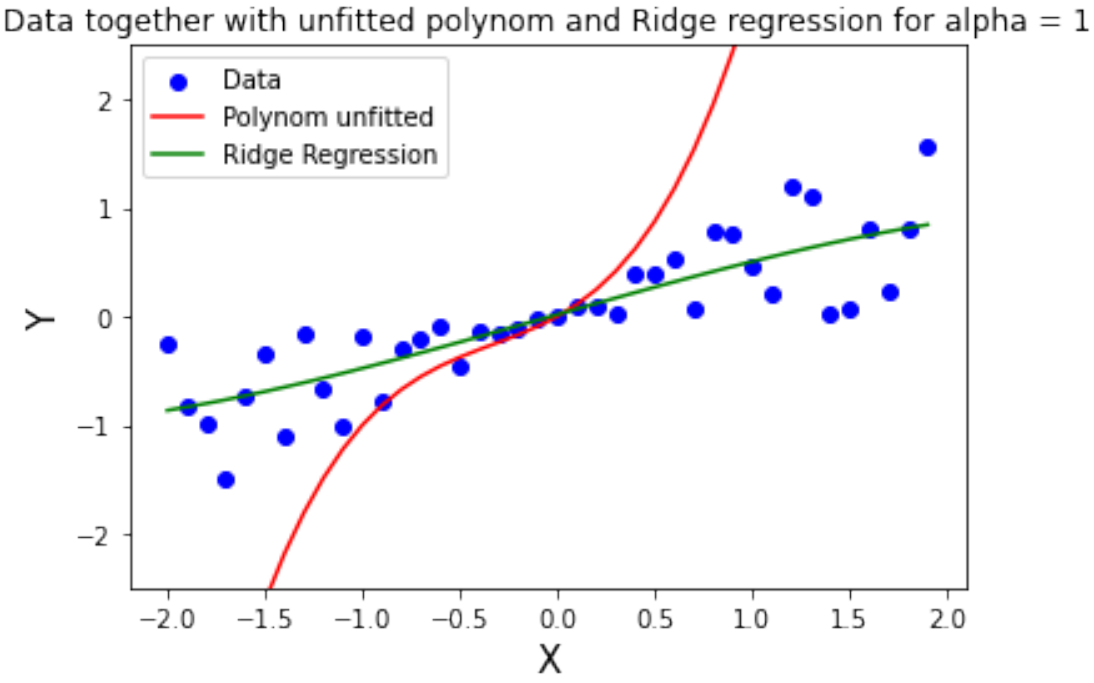
```
plt.plot(x,polynom , c='r', label = "Polynom")

plt.xlabel("X", size = 15)
plt.ylabel("Y", size = 15)
plt.ylim((-2.5,2.5))
plt.title("Data together with unfitted polynom")
plt.legend()
plt.show()

#Apply Ridge regression
Ridge_polynom_fitted = PolynomialFeatures(degree=p, include_bias=True)
Ridge_polynom_features = Ridge_polynom_fitted.fit_transform(x.reshape(-1, 1))

#Use Ridge as fit
Ridge_polynom = Ridge(alpha = 1) #initialise
Ridge_polynom.fit(Ridge_polynom_features, y)
Ridge_values = Ridge_polynom.predict(Ridge_polynom_features)

# plot the results
plt.scatter(x,y, c='b' , label = "Data")
plt.plot(x,polynom, c='red',label = "Polynom unfitted")
plt.plot(x,Ridge_values, c='green', label = "Ridge Regression")
plt.xlabel("X", size = 15)
plt.ylabel("Y", size = 15)
plt.ylim((-2.5,2.5))
plt.title("Data together with unfitted polynom and Ridge regression for alpha =␣
 ↪1")
plt.legend()
plt.show()

#Test different alphas
alphas =  np.arange(0, 2.5, 0.5)
colors = ["forestgreen", "lightgreen", "limegreen", "darkgreen", "green"]

plt.scatter(x,y, c='b' , label = "Data")
plt.plot(x,polynom, c='red',label = "Polynom unfitted")

#Plot Ridge regressions for different alphas
for i in range(np.size(alphas)):
    Ridge_polynom = Ridge(alphas[i])
    Ridge_polynom.fit(Ridge_polynom_features, y)
    Ridge_values = Ridge_polynom.predict(Ridge_polynom_features)
    label_string = "Regression with alpha = ", alphas[i]
    plt.plot(x,Ridge_values, c = colors[i], label = label_string)

plt.xlabel("X", size = 15)
plt.ylabel("Y", size = 15)
```

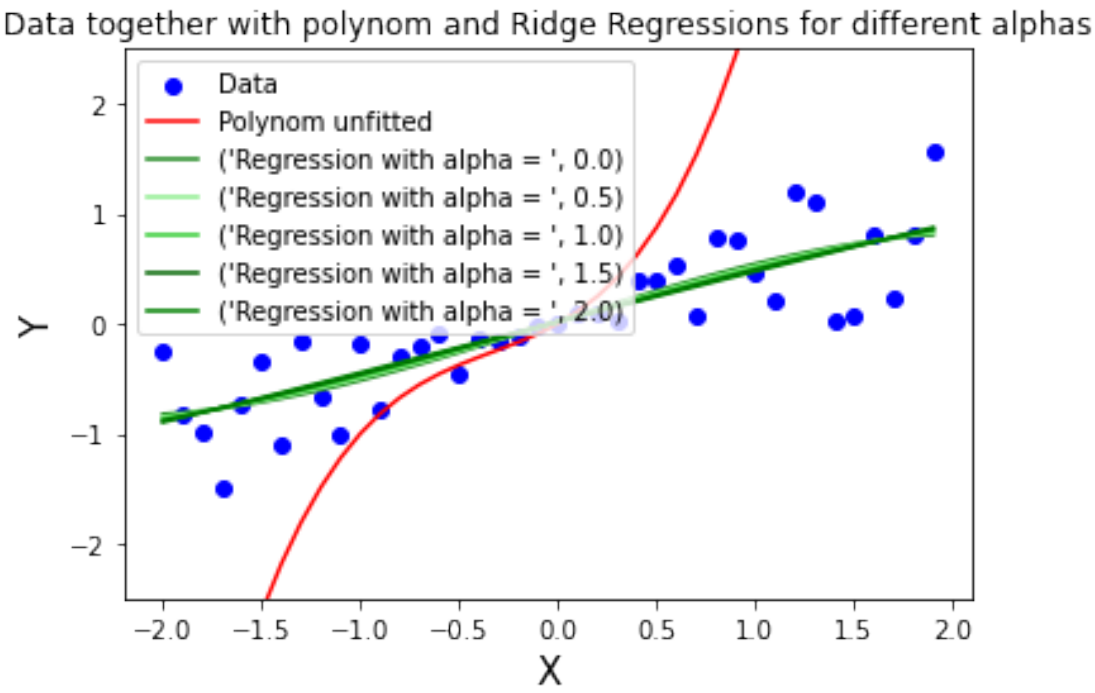
```
plt.ylim((-2.5,2.5))
plt.title("Data together with polynom and Ridge Regressions for different␣
 ↪alphas")
plt.legend()
plt.show()
```

```
Degree of the polynom: 3
```




Data together with unfitted polynom
Data
Polynom
Y
X
2
1
0
−1
−2
−2.0
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0


Data together with unfitted polynom and Ridge regression for alpha = 1
Data
Polynom unfitted
Ridge Regression
Y
X
2
1
0
−1
−2
−2.0
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0


Data together with polynom and Ridge Regressions for different alphas
Data
Polynom unfitted
('Regression with alpha = ', 0.0)
('Regression with alpha = ', 0.5)
('Regression with alpha = ', 1.0)
('Regression with alpha = ', 1.5)
('Regression with alpha = ', 2.0)
Y
X
2
1
0
−1
−2
−2.0
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0

[ ]: