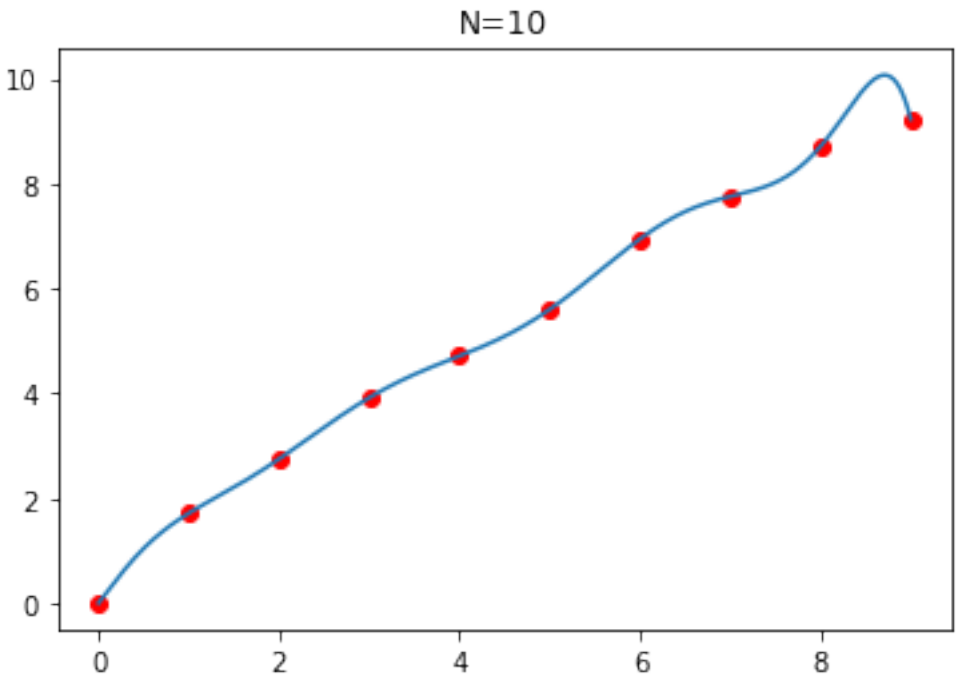
overfitting

October 11, 2022

[1]:
```
import numpy as np
from scipy import interpolate
from sklearn.linear_model import Ridge, LinearRegression
from sklearn.preprocessing import PolynomialFeatures
from sklearn.pipeline import make_pipeline
from matplotlib import pyplot as plt
```

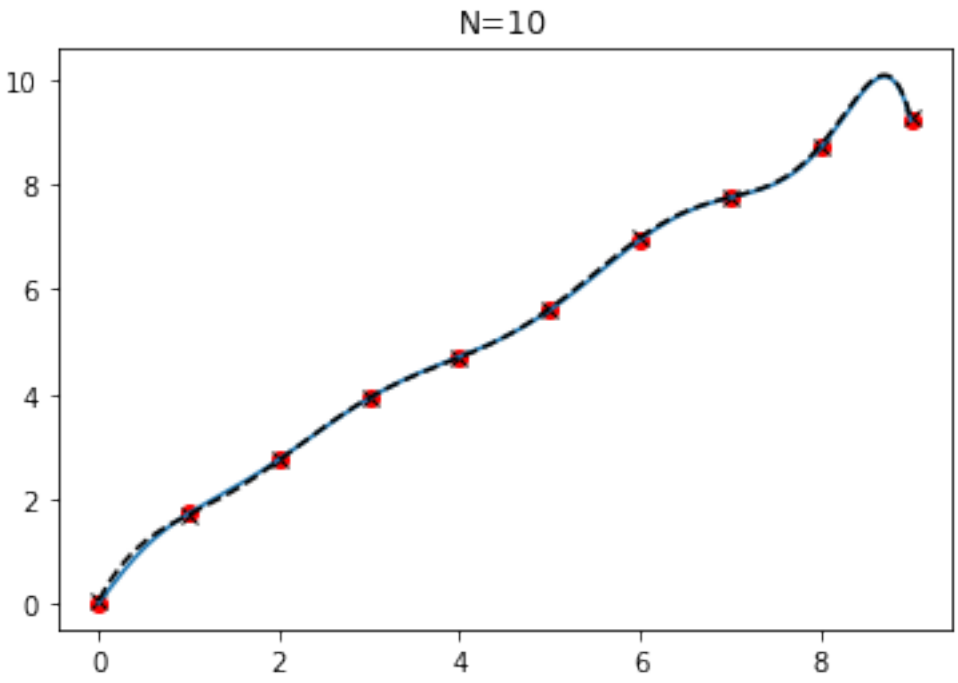
[2]:
```
n = 10
x0 = np.arange(0, n)
y0 = x0 + np.random.rand(n)

x = np.linspace(0,n-1,1000)

f0 = interpolate.BarycentricInterpolator(x0, y0)

plt.figure()
plt.plot(x0, y0, 'ro')
plt.plot(x, f0(x))

plt.title('N={:d}'.format(n))
```

[2]:
```
Text(0.5, 1.0, 'N=10')
```



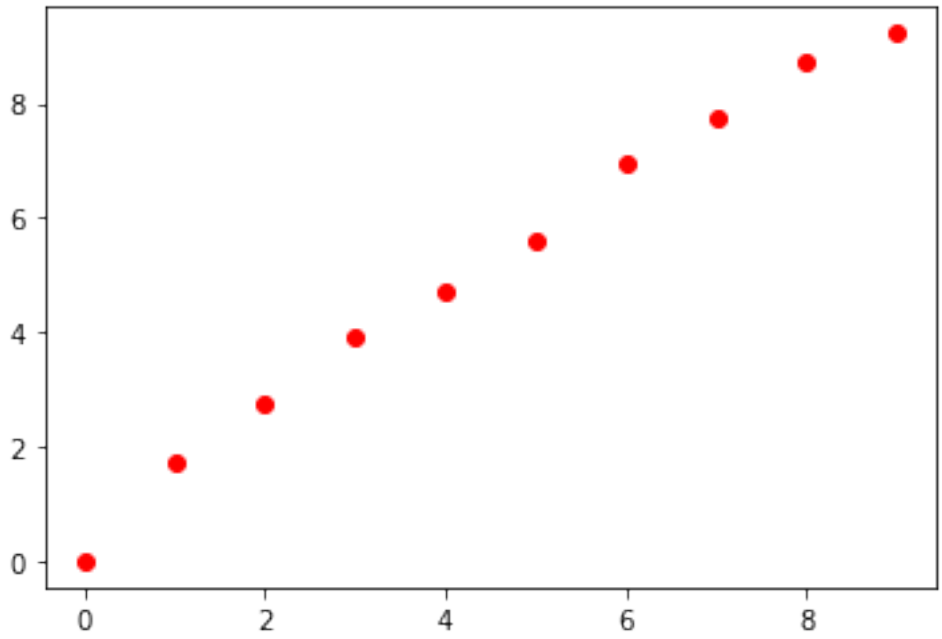
[3]:
```
eps=1e-1
y0_noise = eps*(np.random.rand(n)-0.5) + y0

f1 = interpolate.BarycentricInterpolator(x0, y0_noise)

plt.figure()
plt.plot(x0, y0, 'ro')
plt.plot(x0, y0_noise, 'kx')
plt.plot(x, f0(x))
plt.plot(x, f1(x), 'k--')

plt.title('N={:d}'.format(n))
```

[3]:
```
Text(0.5, 1.0, 'N=10')
```



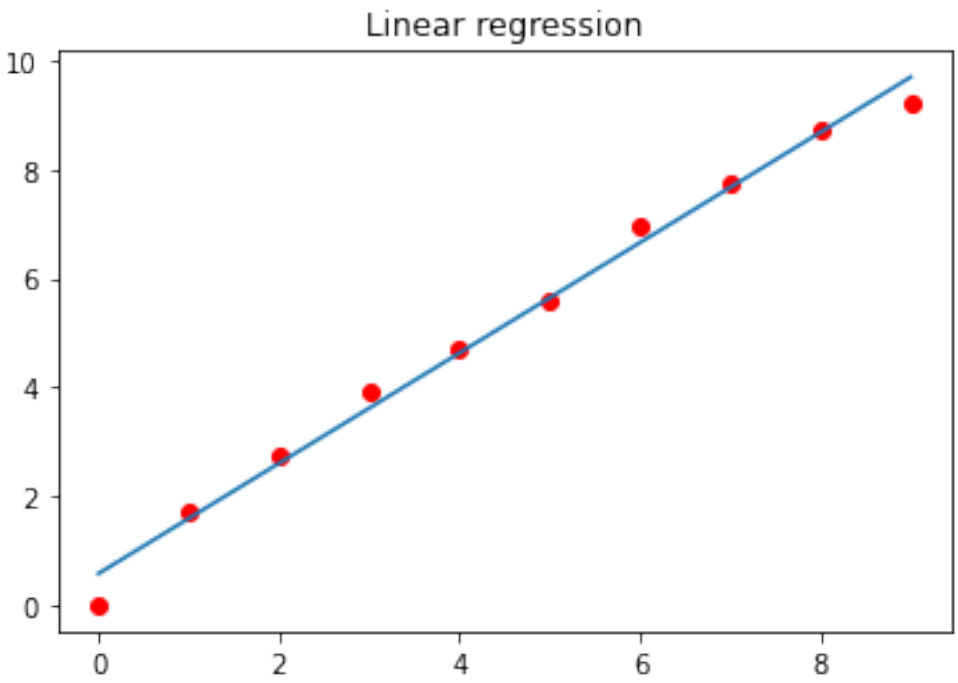
[4]:
```
plt.figure()
plt.plot(x0, y0, 'ro')
plt.title(' ')
```

[4]: Text(0.5, 1.0, ' ')



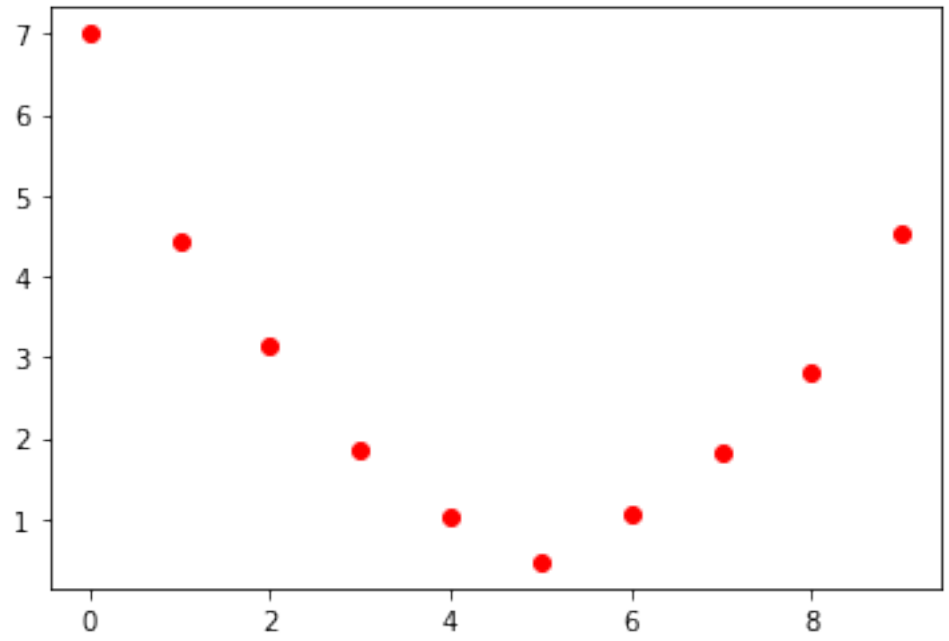
[5]:
```
f2 = LinearRegression().fit(x0.reshape(-1, 1), y0)

plt.figure()
plt.plot(x0, y0, 'ro')
plt.plot(x, f2.predict(x.reshape(-1, 1)))
plt.title('Linear regression')
```

[5]:
```
Text(0.5, 1.0, 'Linear regression')
```



[6]:
```
y1 = (0.5*(x0-n/2))**2 + np.random.rand(n)
plt.plot(x0, y1, 'ro')
```

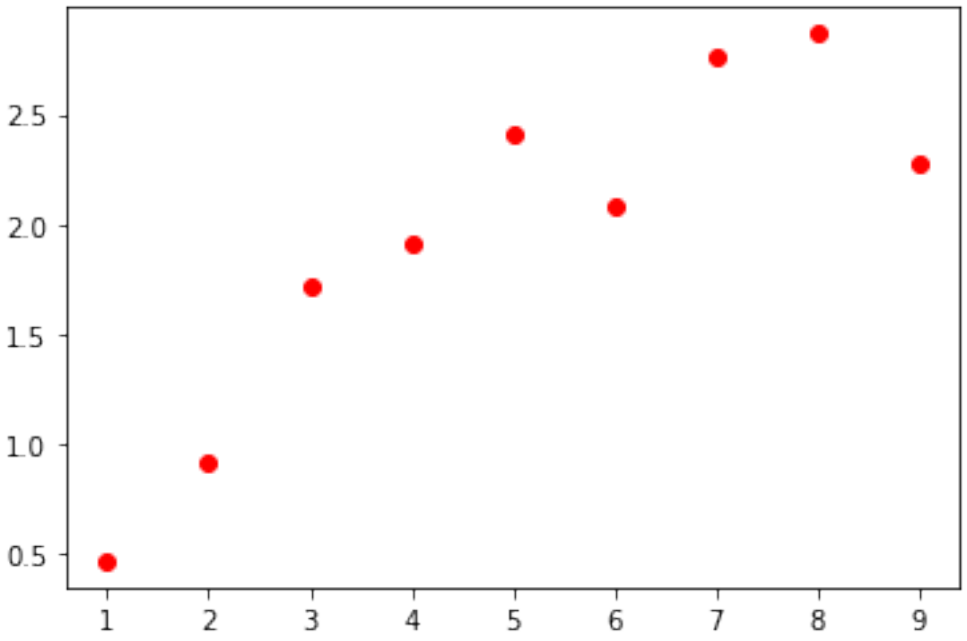
[6]: [<matplotlib.lines.Line2D at 0x1df6d4f81c0>]



[7]:
```
y2 = np.log(x0) + np.random.rand(n)
plt.plot(x0, y2, 'ro')
```

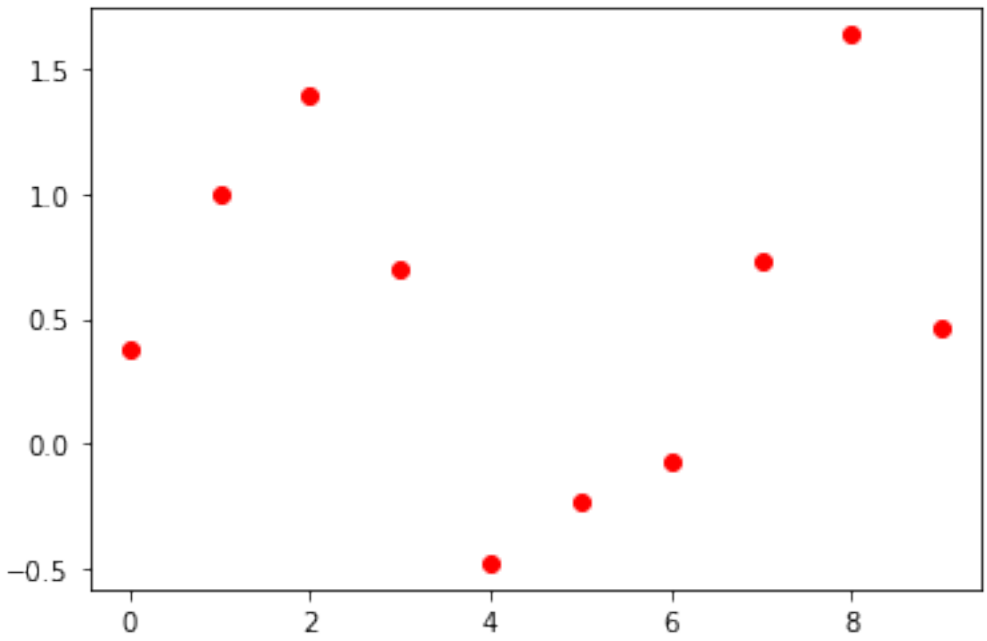
```
<ipython-input-7-ed639471062e>:1: RuntimeWarning: divide by zero encountered in
log
  y2 = np.log(x0) + np.random.rand(n)
```

[7]: [<matplotlib.lines.Line2D at 0x1df6d548a30>]



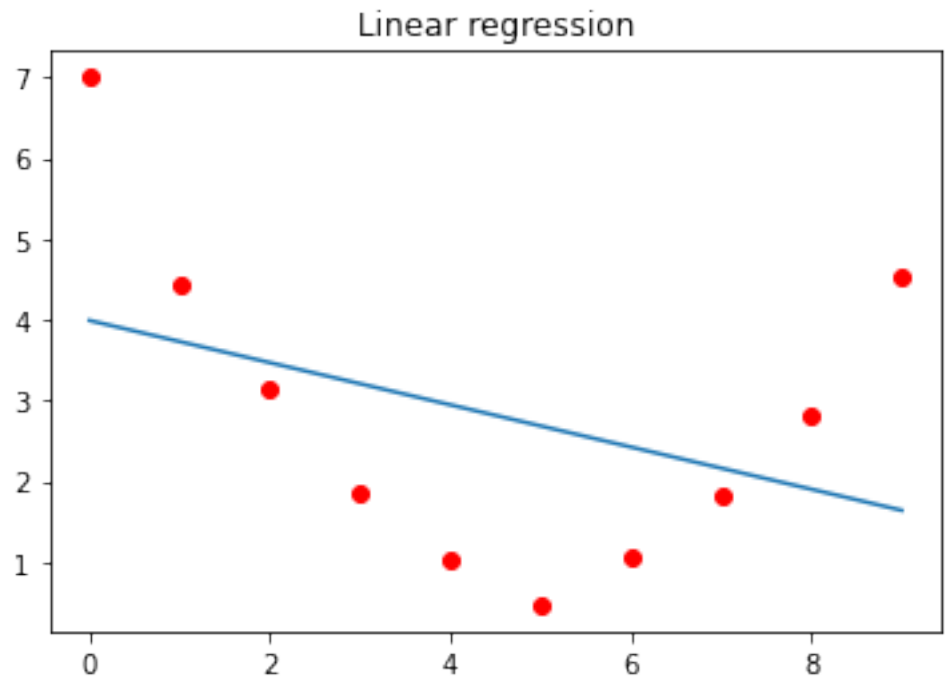
[8]:
```
y3 = np.sin(x0) + np.random.rand(n)
plt.plot(x0, y3, 'ro')
```

[8]: [<matplotlib.lines.Line2D at 0x1df6d5a6850>]



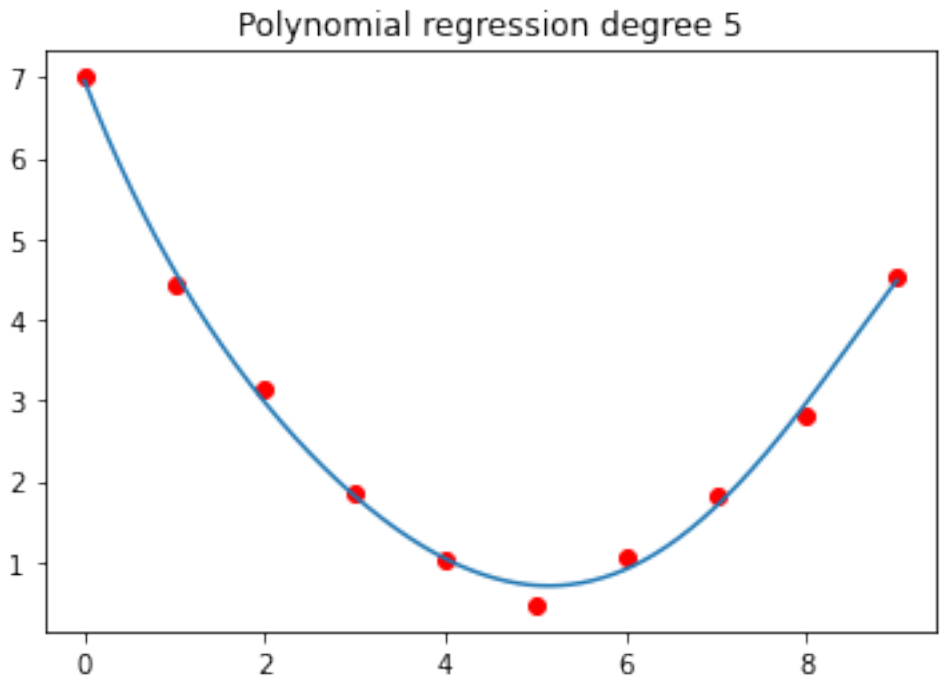
[9]:
```
f3 = LinearRegression().fit(x0.reshape(-1, 1), y1)

plt.figure()
plt.plot(x0, y1, 'ro')
plt.plot(x, f3.predict(x.reshape(-1, 1)))
plt.title('Linear regression')
```

[9]:
```
Text(0.5, 1.0, 'Linear regression')
```



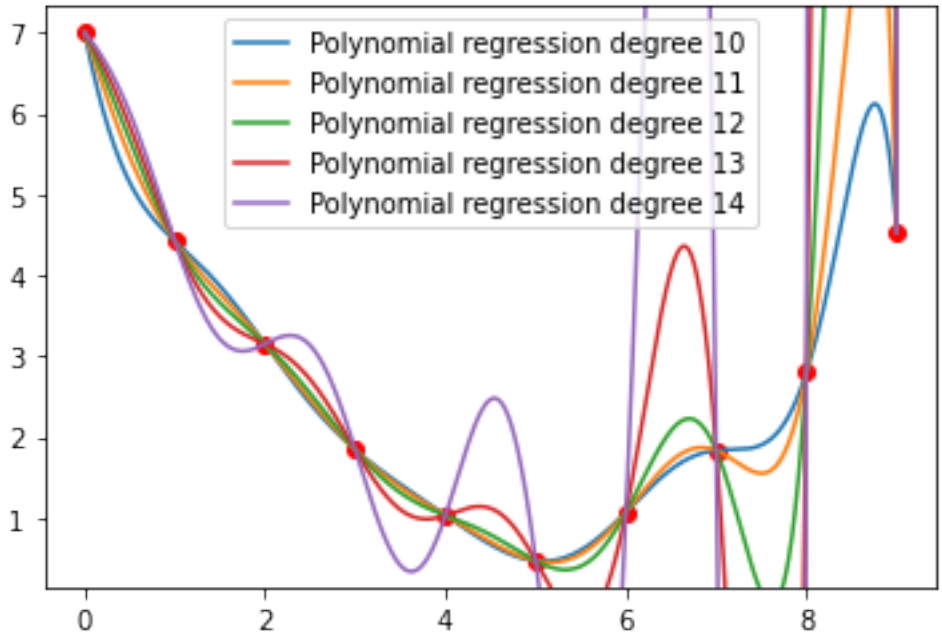
[10]:
```
degree = 5
f4 = make_pipeline(PolynomialFeatures(degree), LinearRegression())␣
 ↪#Ridge(alpha=1e-3)
f4.fit(x0.reshape(-1, 1), y1)

plt.figure()
plt.plot(x0, y1, 'ro')
plt.plot(x, f4.predict(x.reshape(-1, 1)))
plt.title('Polynomial regression degree {:d}'.format(degree))
```

[10]:
```
Text(0.5, 1.0, 'Polynomial regression degree 5')
```



[11]:
```
plt.figure()
plt.plot(x0, y1, 'ro')
ylim = plt.ylim()

for degree in [10, 11, 12, 13, 14]:
    f4 = make_pipeline(PolynomialFeatures(degree), LinearRegression())␣
 ↪#Ridge(alpha=1e-3)
    f4.fit(x0.reshape(-1, 1), y1)

    plt.plot(x, f4.predict(x.reshape(-1, 1)), label='Polynomial regression␣
 ↪degree {:d}'.format(degree))
plt.ylim(ylim)
plt.legend()
```

[11]: <matplotlib.legend.Legend at 0x1df6d6985e0>


Polynomial regression degree 10
Polynomial regression degree 11
Polynomial regression degree 12
Polynomial regression degree 13
Polynomial regression degree 14
7
6
5
4
3
2
1
0
2
4
6
8

[ ]: