Introduction to Computer Programming

Week 9.2: Curve Fitting

In [43]:
```
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
```

Fitted function

Example 1:

Fit a first degree polynomial (linear function) to the `x,y` data.

Print the coefficients of the fitted function.

In [44]:
```
x = np.array([1, 6, 3, 4, 10, 2, 7, 8, 9, 5])
y = np.array([2, 4, 5, 4, 13, 3, 4, 8, 12, 4])

c1 = np.polyfit(x, y, 1) # coefficients of 1st degree fitted poly

print(c1)
print(c1[0], c1[1])
```

```
[ 1.07272727e+00 -5.61733355e-15]
1.0727272727272728 -5.617333549722272e-15
```

In [45]:
```
from numpy.polynomial import Polynomial

c = Polynomial.fit(x,y,1)
print(c)
```

```
poly([5.9        4.82727273])
```

Try it yourself

Example 2:

Fit a second degree polynomial to the `x,y` data.
(Remember to import numpy to use `polyfit`).

Print the coefficients of the fitted function.

In [46]:
```
x = np.array([1, 6, 3, 4, 10, 2, 7, 8, 9, 5])
y = np.array([2, 4, 5, 4, 13, 3, 4, 8, 12, 4])

c2 = np.polyfit(x, y, 2) # 2nd degree poly

print(c2[0], c2[1], c2[2])
```

```
0.19318181818181812 -1.0522727272727261 4.249999999999994
```

Fitted data

Example 3:

Use `numpy.polyval` to generate x,y data of the fitted linear function.

In [47]:
```
x_new = np.array(sorted(x))        # x values, sorted monotonically fo

yfit1 = np.polyval(c1, x_new) # 1st degree polynomial
```

Try it yourself

Example 4:

Use `numpy.polyval` to generate x,y data of the fitted second degree polynomial function.

In [48]:
```
x_new = np.array(sorted(x))        # x values, sorted monotonically fo

yfit2 = np.polyval(c2, x_new) # 2nd degree polynomial
```

Plotting fitted data

Example 5: Plot the raw data as a scatter plot and fitted linear function as a line graph ont eh same figure.

In [49]:

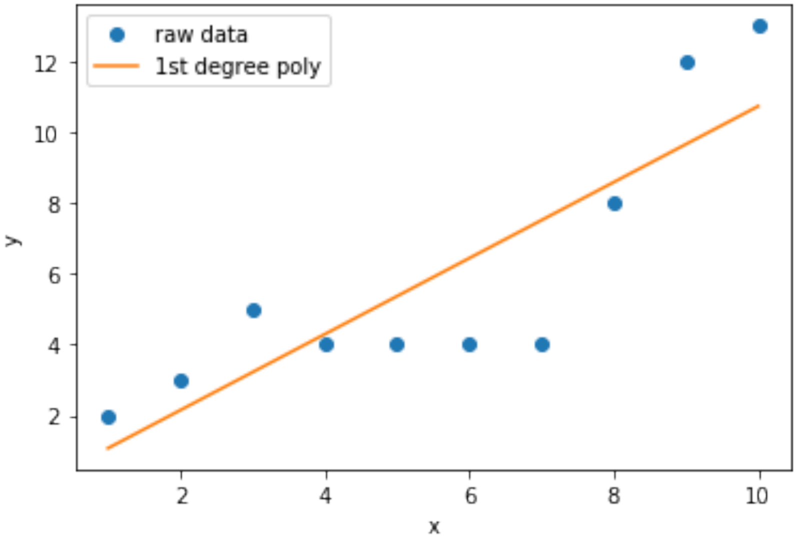
```
# plot data
plt.plot(x, y, 'o',     label='raw data')            # raw data
plt.plot(x_new, yfit1, label='1st degree poly');  # fitted 1st degree

plt.legend()

plt.xlabel('x')
plt.ylabel('y')
```

Out[49]:

```
Text(0, 0.5, 'y')
```



Try it yourself

Example 6:
Plot the raw data as a scatter plot and second degree polynomial function as a line graph on the same figure.

In [50]:

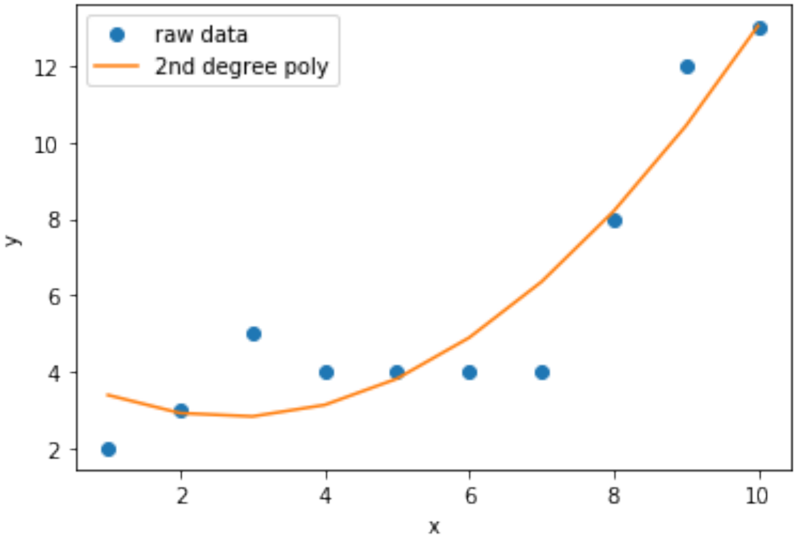
```
# plot data
plt.plot(x, y, 'o',     label='raw data')          # raw data
#plt.plot(x_new, yfit1, label='1st degree poly');  # fitted 1st degree
plt.plot(x_new, yfit2, label='2nd degree poly');  # fitted 2nd degree

plt.legend()

# label the axes
plt.xlabel('x')
plt.ylabel('y')
```

Out[50]:

```
Text(0, 0.5, 'y')
```



Example 7

Fit the function $y = ae^{bx}$ which we defined earlier as `exponential` and find the RMSE:

In [51]:

```
from scipy.optimize import curve_fit

def RMSE(x, y, yfit):
    "Returns the RMSE of a y data fitted to x-y raw data"
    # error
    e = (yfit - y)

    # RMSE
    return np.sqrt(np.sum(e**2)/ len(x))

def exponential(x, a, b): # input arguments are independent variable,
    y = a * np.exp(b*x)
    return y
```

In [52]:

```
c, cov = curve_fit(exponential, x, y)  # constants of fitted function
yfit = exponential(x, *c)              # no need to sort x monotonical
rmse = RMSE(x,y,yfit)                  # goodness of fit
print(f'RMSE = {rmse}')
```

```
RMSE = 1.3338248760975626
```

In [ ]:

In [ ]: