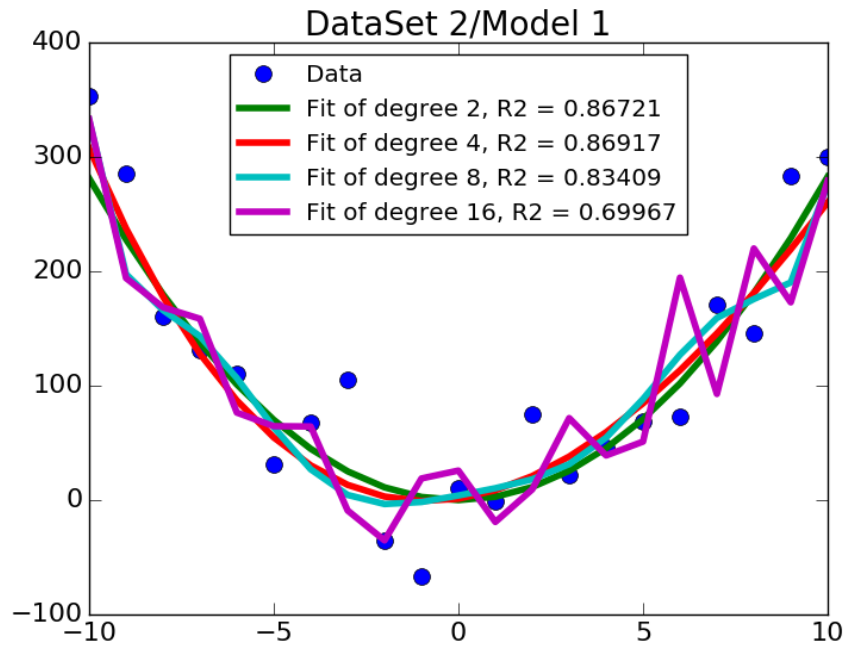
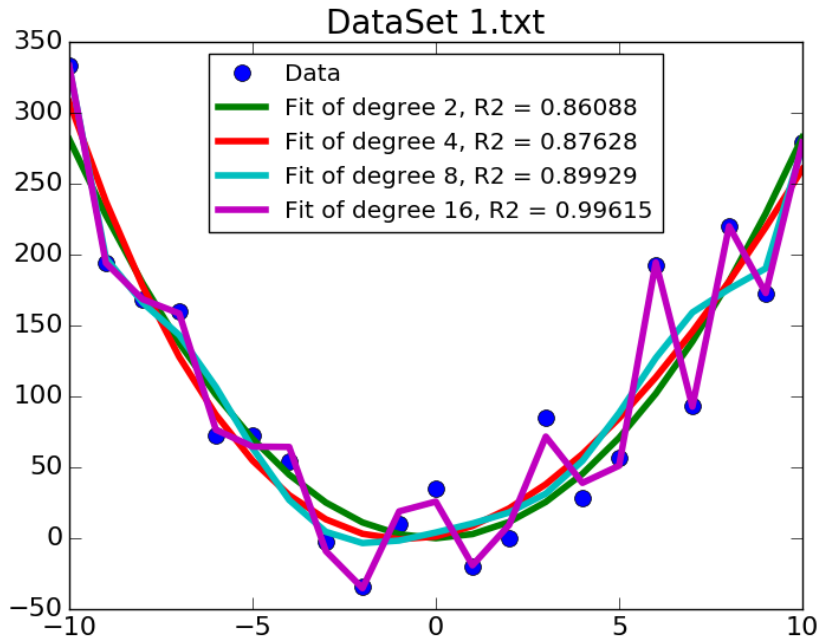


# Understanding Experimental Data, cont.

---

# Training and Testing Errors



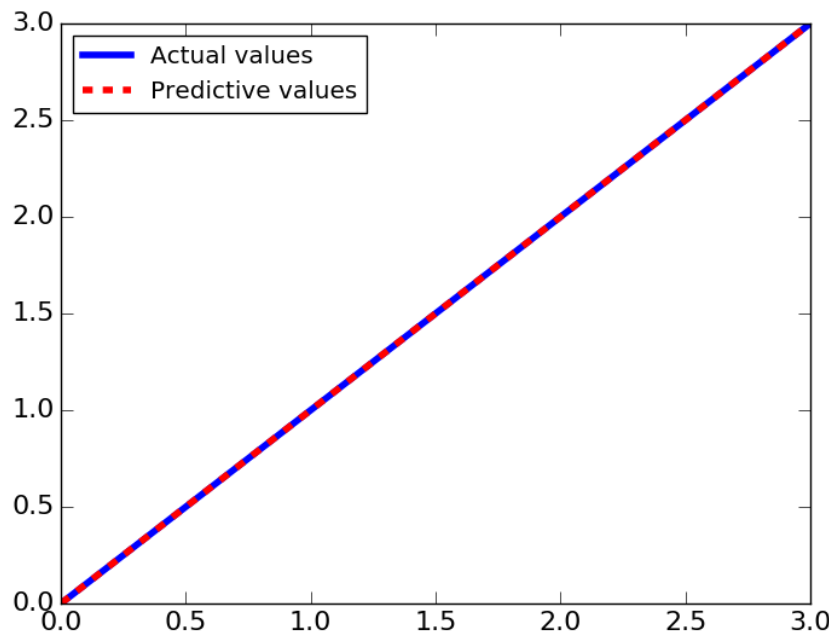
# Increasing the Complexity

---

- What happens when we increase order of polynomial?
  - Can we get a worse fit to training data?
- If extra term is useless, coefficient will merely be zero
- But if data is noisy, can fit the noise rather than the underlying pattern in the data

# Fitting a Quadratic to a Perfect Line

```
xVals = (0,1,2,3)
yVals = xVals
pylab.plot(xVals, yVals, label = 'Actual values')
a,b,c = pylab.polyfit(xVals, yVals, 2)
print('a =', round(a, 4), 'b =', round(b, 4),
      'c =', round(c, 4))
estYVals = pylab.polyval((a,b,c), xVals)
pylab.plot(xVals, estYVals, 'r--', label = 'Predictive values')
print('R-squared = ', rSquared(yVals, estYVals))
```



$$y = ax^2 + bx + c$$

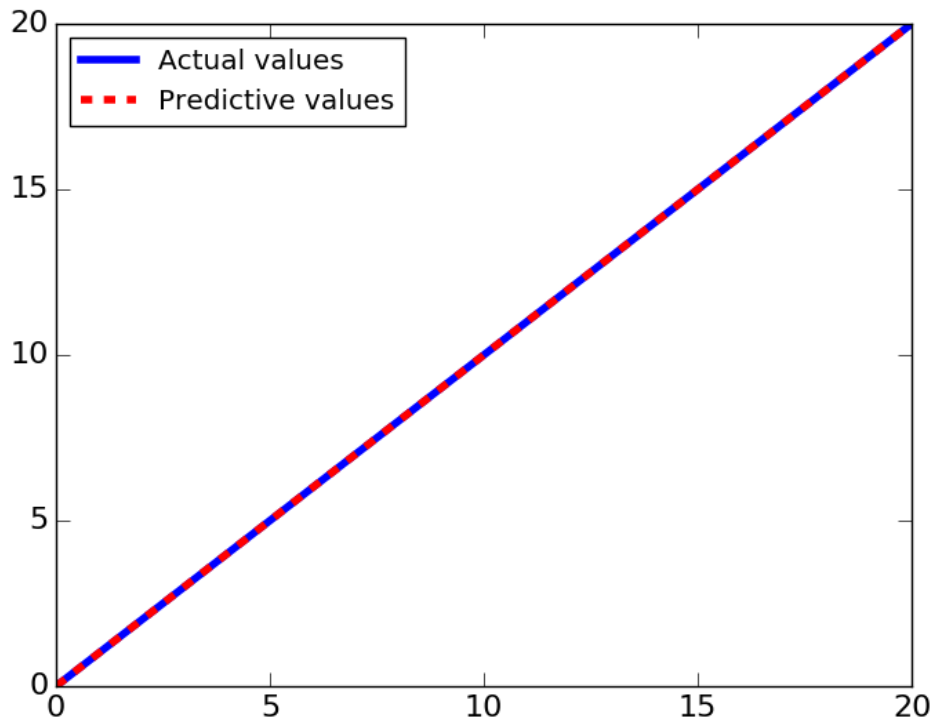
$$y = 0x^2 + 1x + 0$$

$$y = x$$

# Predict Another Point Using Same Model

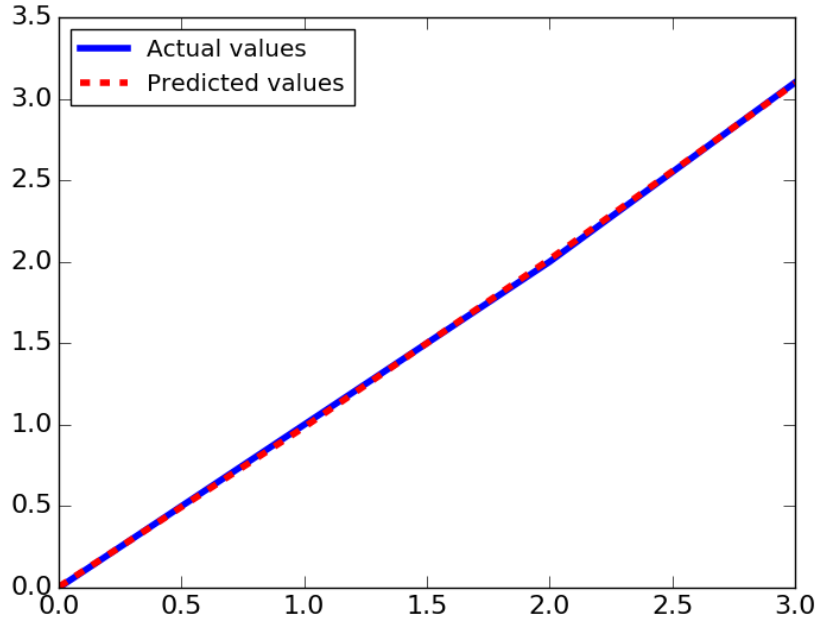
---

```
xVals = xVals + (20,)  
yVals = xVals  
pylab.plot(xVals, yVals, label = 'Actual values')  
estYVals = pylab.polyval((a,b,c), xVals)  
pylab.plot(xVals, estYVals, 'r--', label = 'Predictive values')  
print('R-squared = ', rSquared(yVals, estYVals))
```



# Simulate a Small Measurement Error

```
xVals = (0,1,2,3)
yVals = (0,1,2,3.1)
pylab.plot(xVals, yVals, label = 'Actual values')
model = pylab.polyfit(xVals, yVals, 2)
print(model)
estYVals = pylab.polyval(model, xVals)
pylab.plot(xVals, estYVals, 'r--', label = 'Predicted values')
print('R-squared = ', rSquared(yVals, estYVals))
```



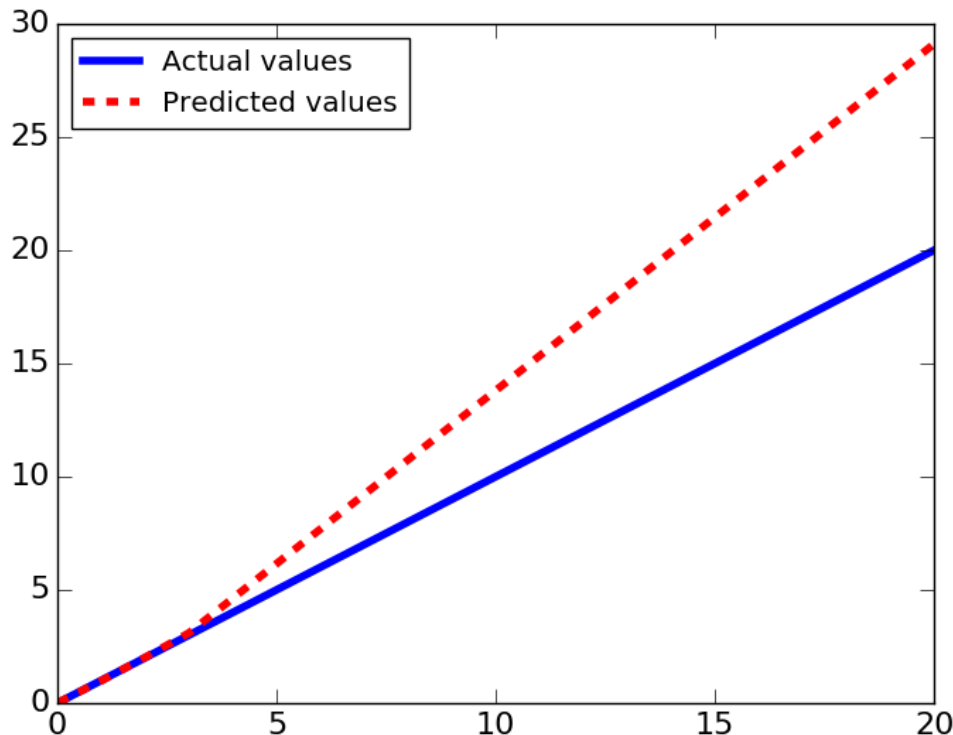
$$y = ax^2 + bx + c$$

$$y = .025x^2 + .955x + .005$$

$$R\text{-squared} = \underline{0.9994}$$

# Predict Another Point Using Same Model

```
xVals = xVals + (20,)
yVals = xVals
estYVals = pylab.polyval(model, xVals)
print('R-squared = ', rSquared(yVals, estYVals))
pylab.figure()
pylab.plot(xVals, estYVals)
```

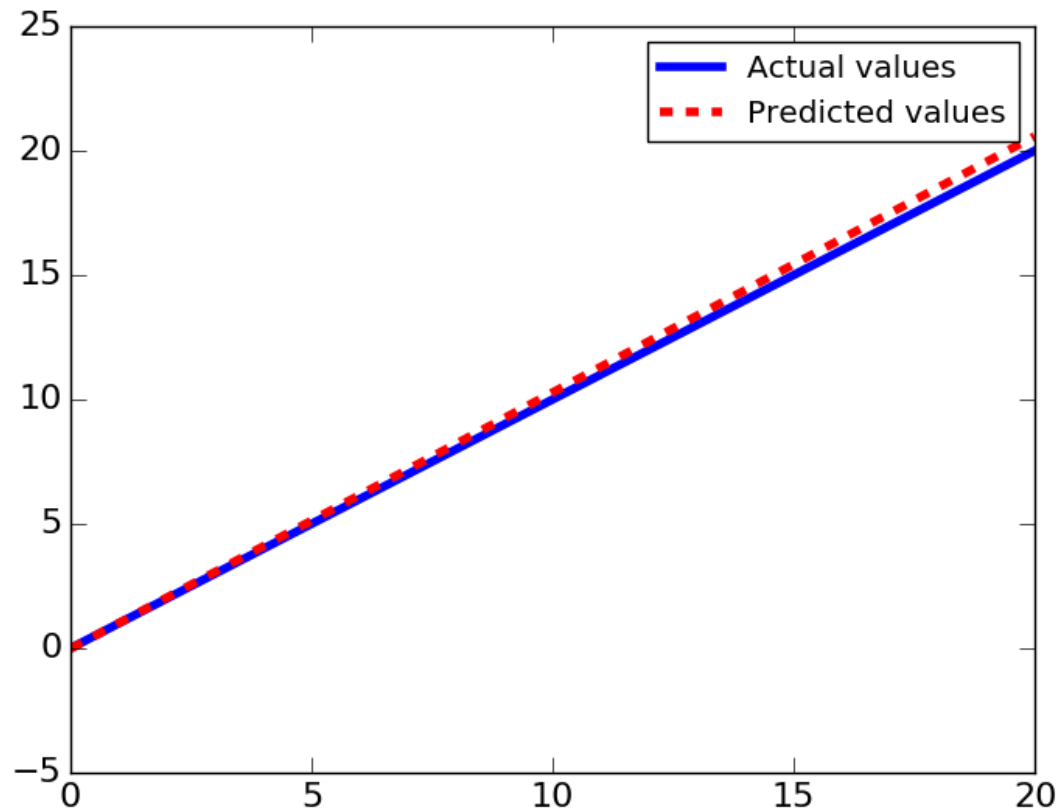


R-squared = 0.7026

# Suppose We Had Used a First-degree Fit

---

■ `model = pylab.polyfit(xVals, yVals, 1)`





# The Take Home Message

---

- Choosing an overly-complex model leads to **overfitting** to the training data
- Increases the risk of a model that works **poorly** on data not included in the training set
- On the other hand choosing an **insufficiently** complex model has other problems
  - As we saw when we fit a line to data that was basically parabolic