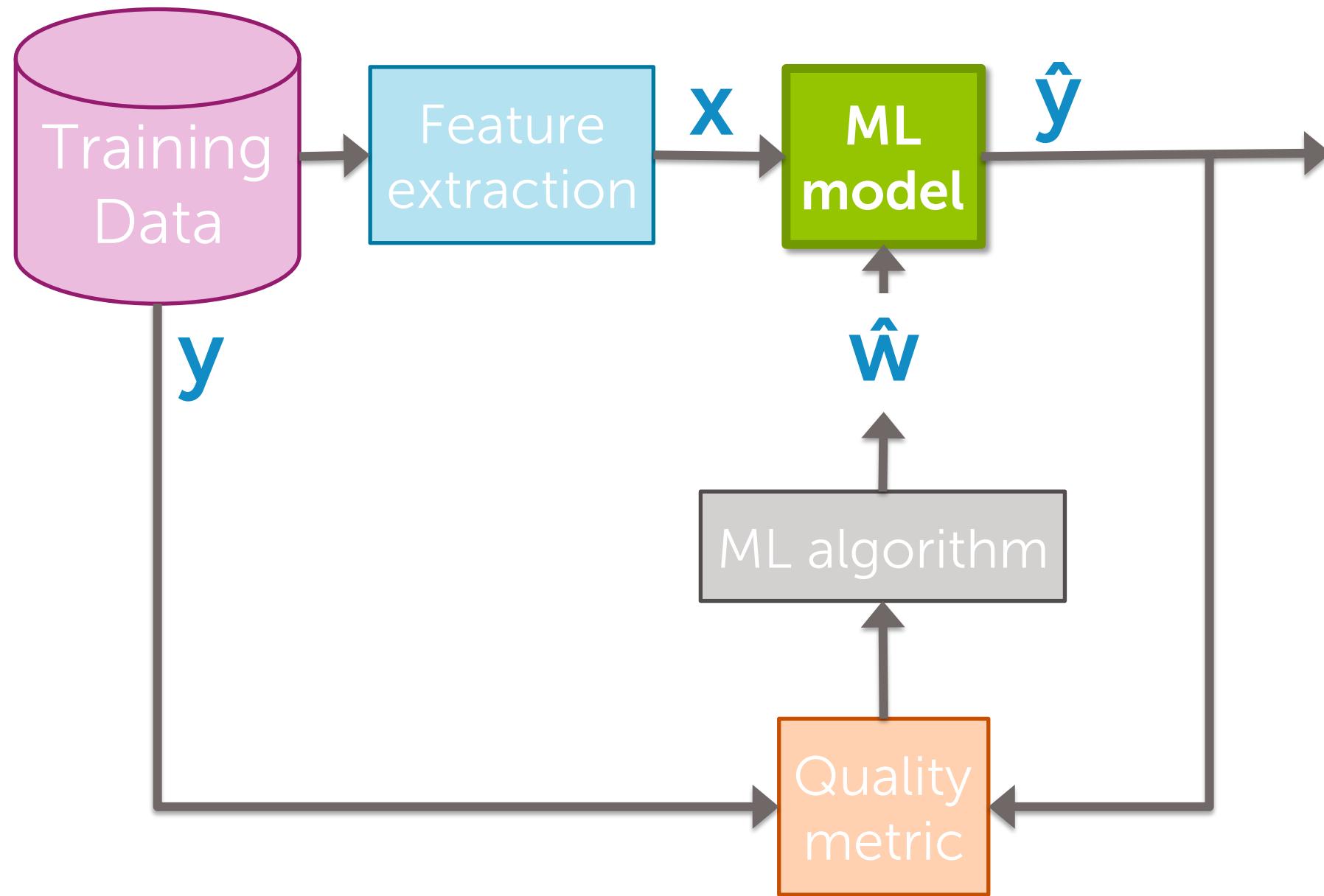
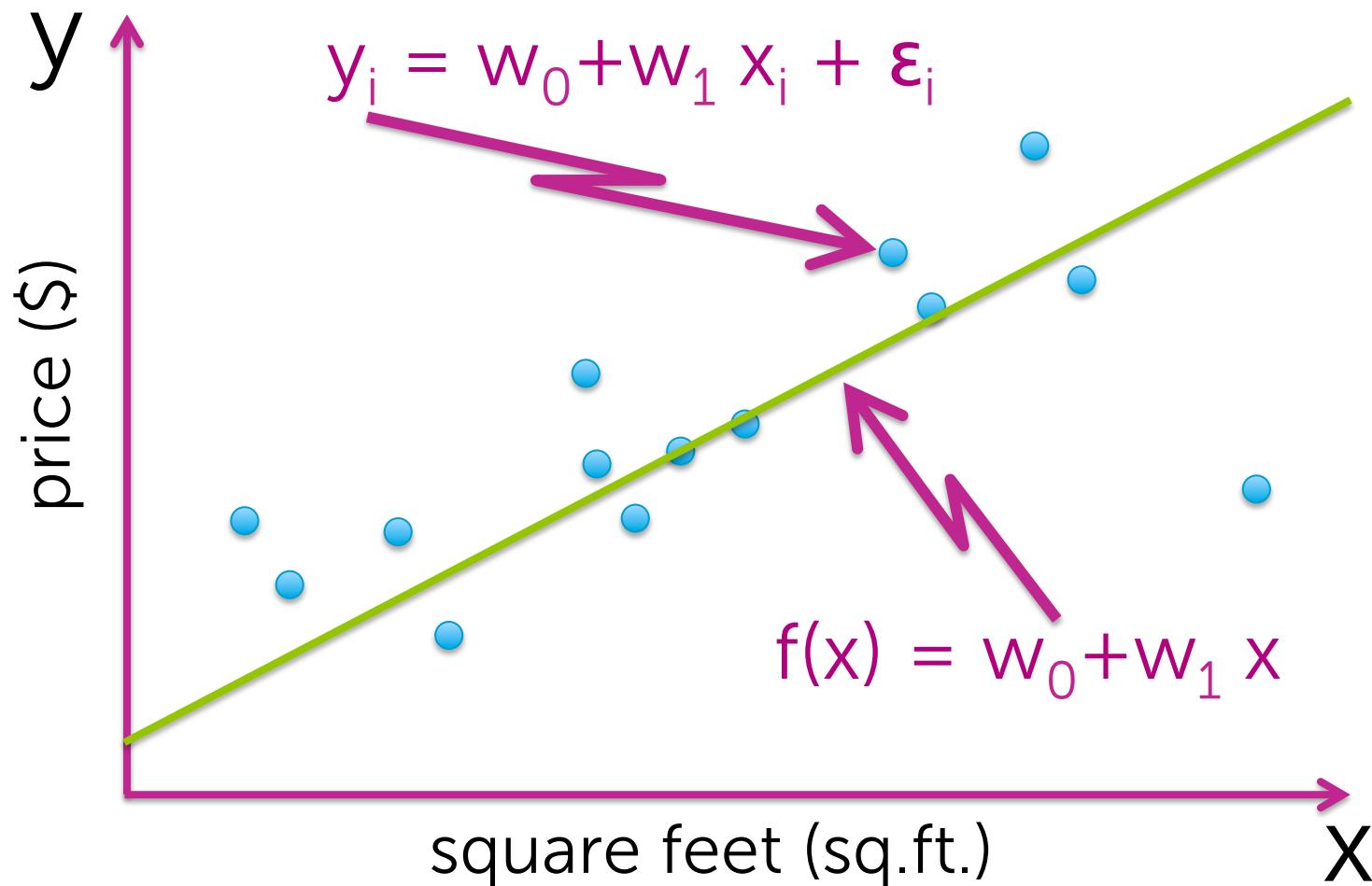


Multiple Regression: Linear regression with multiple features

Emily Fox & Carlos Guestrin
Machine Learning Specialization
University of Washington



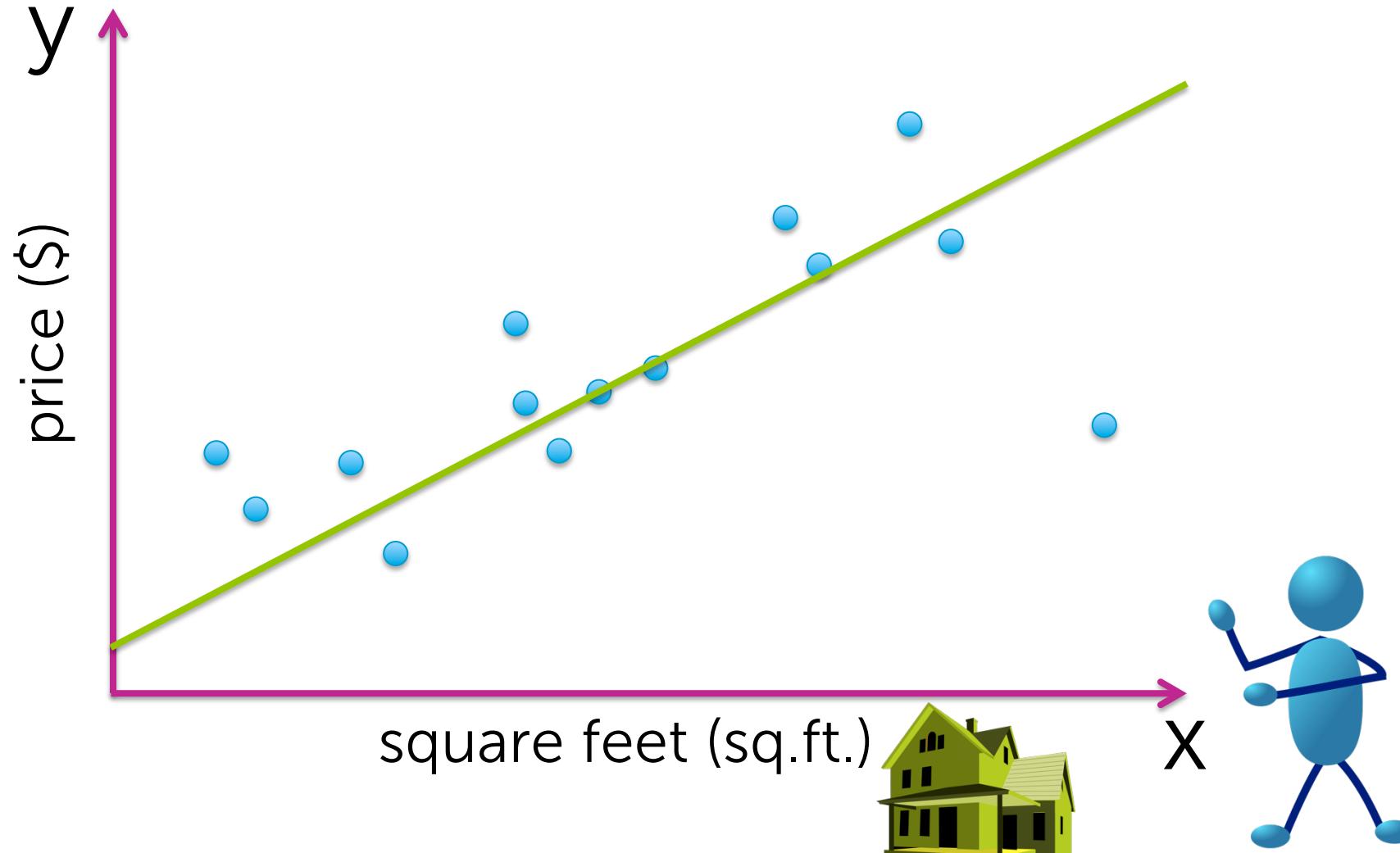
Simple linear regression model



More complex functions of a single input

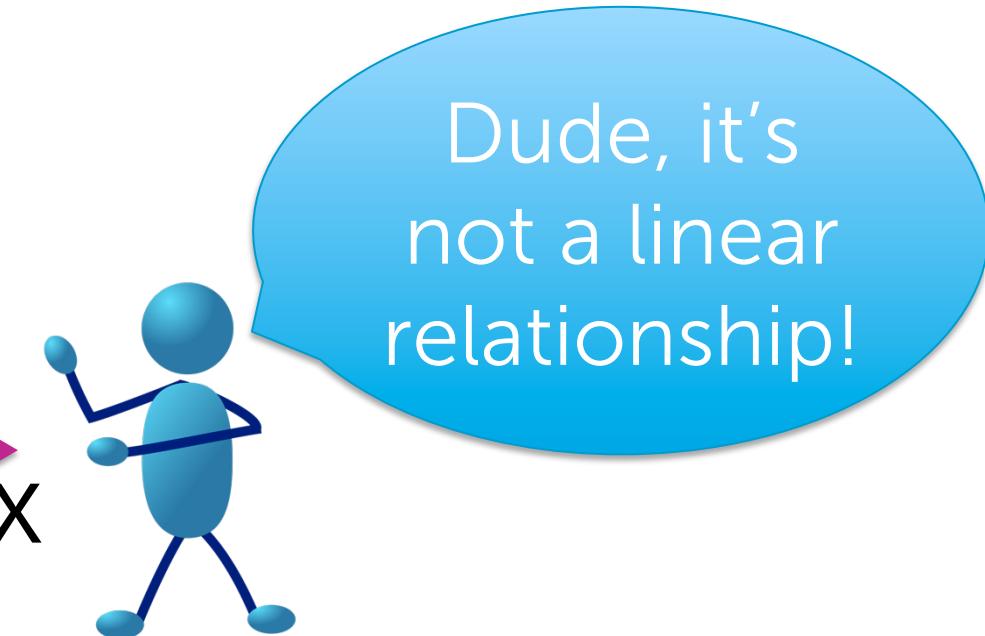
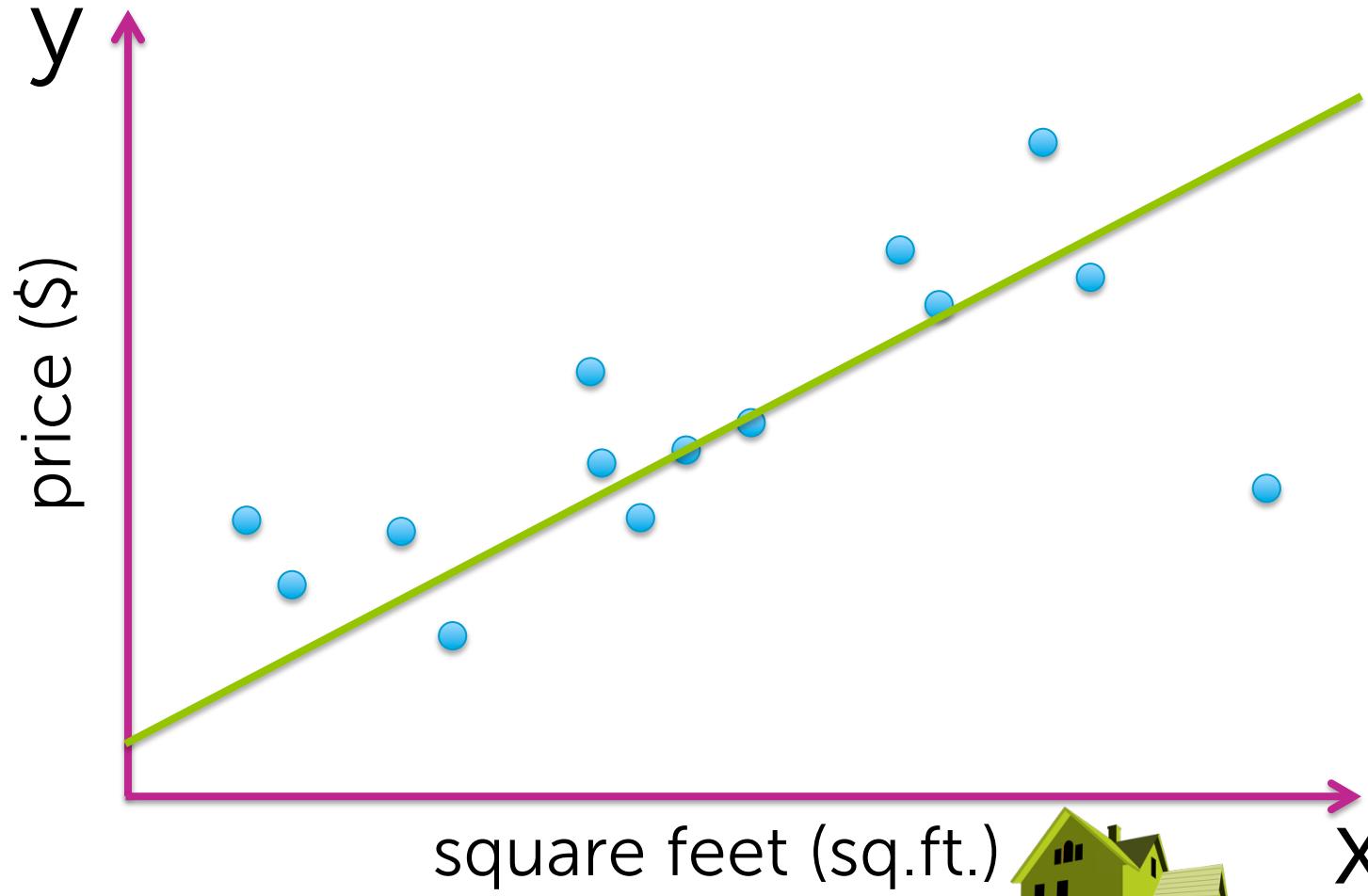
Polynomial regression

Fit data with a line or ... ?

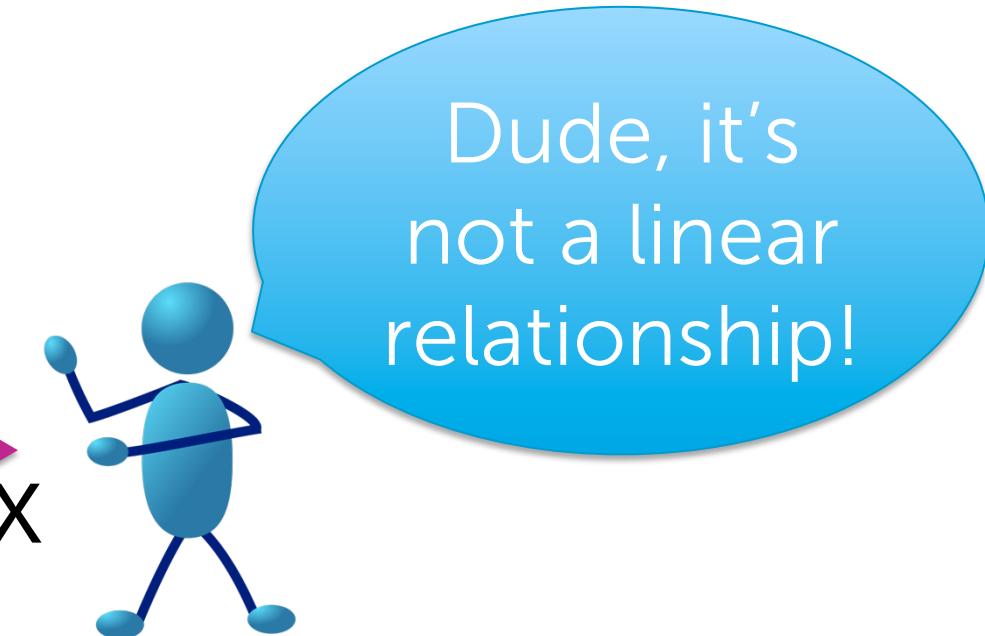
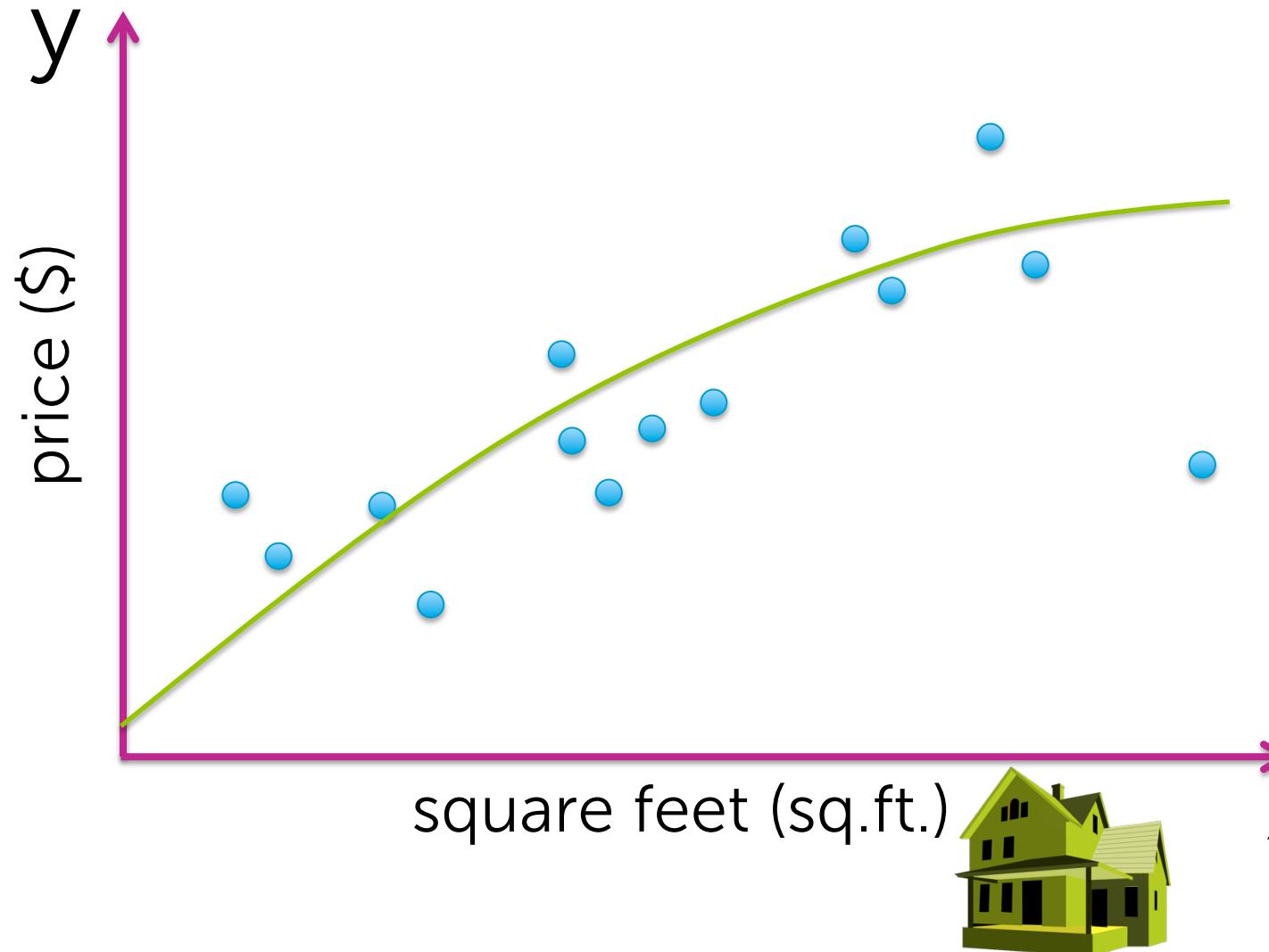


You show
your friend
your analysis

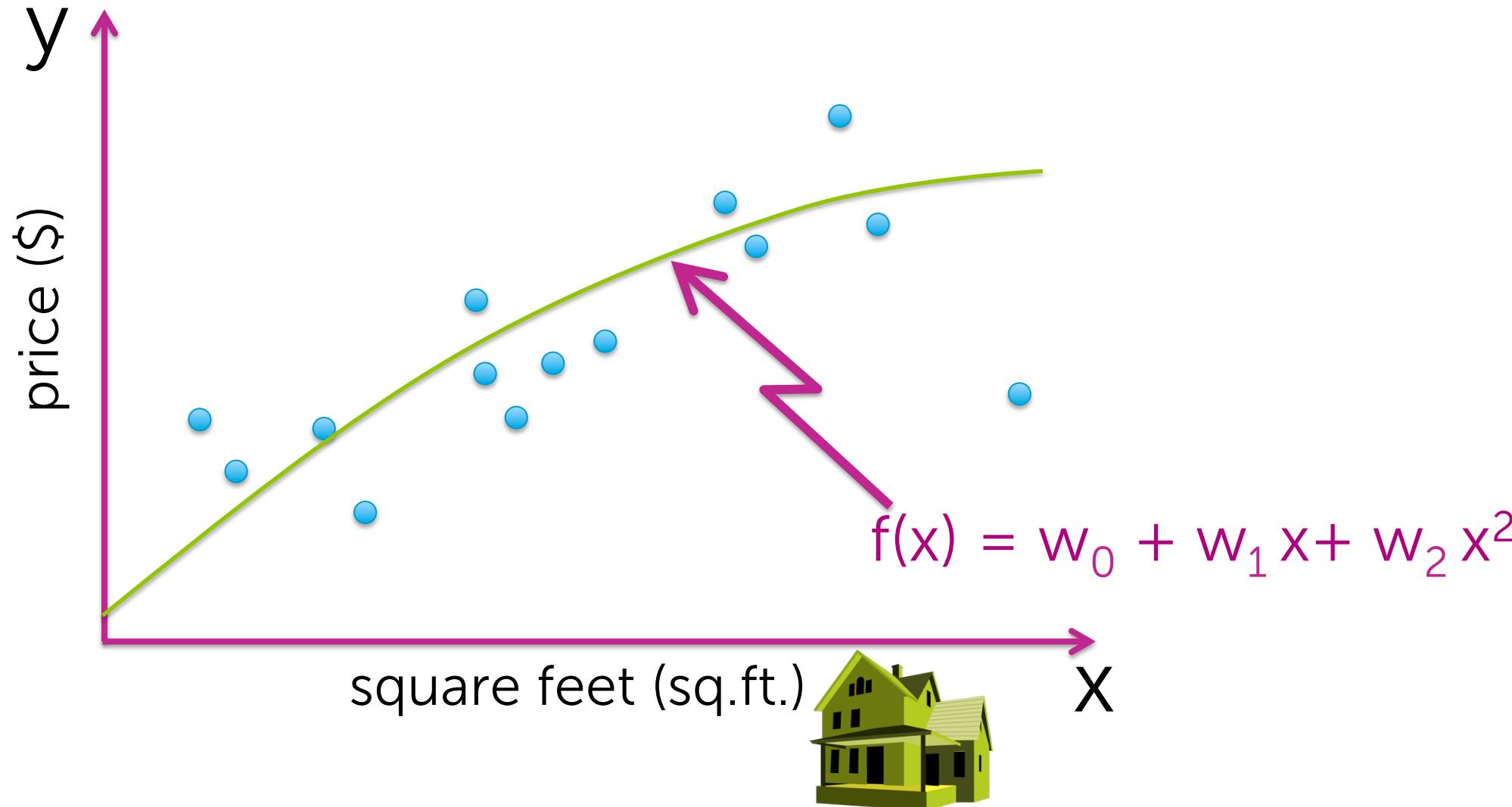
Fit data with a line or ... ?



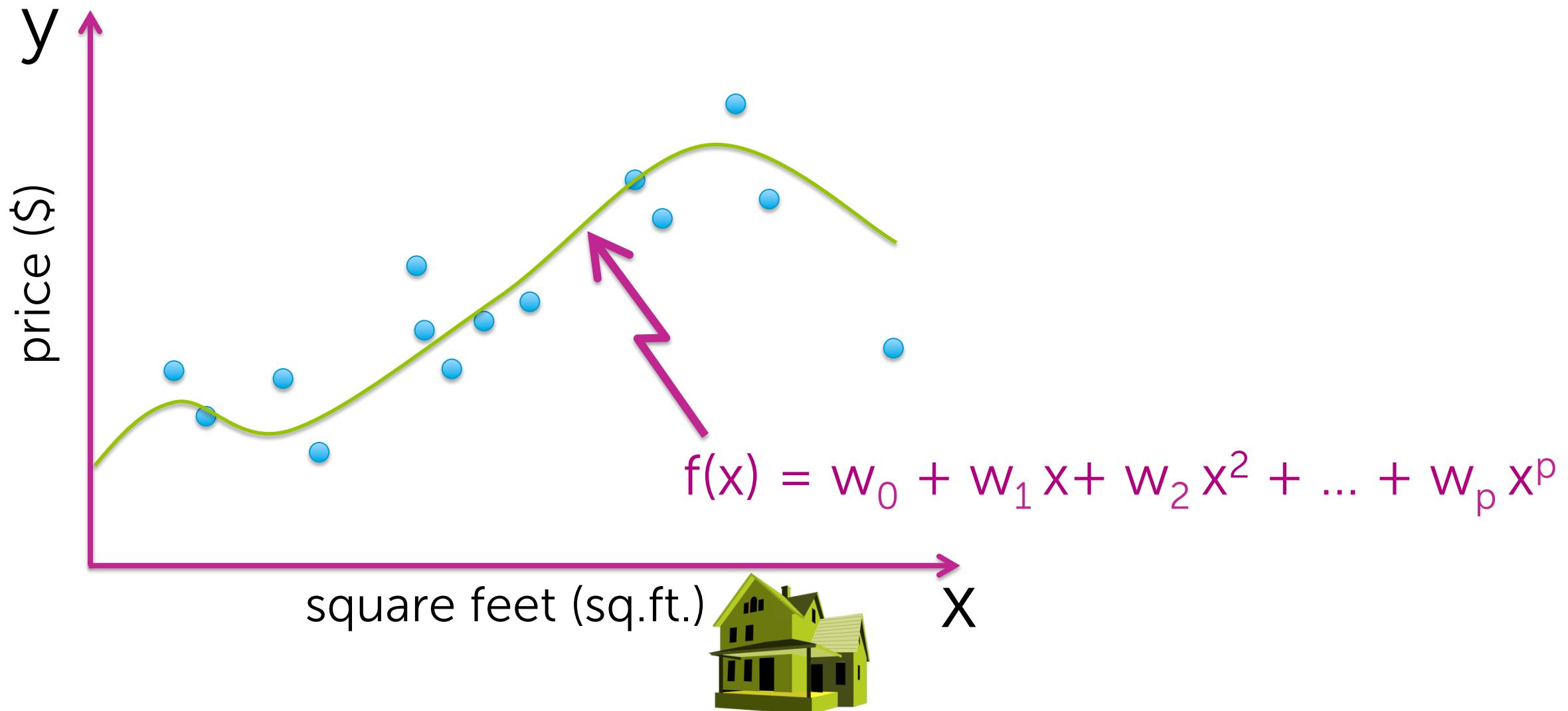
What about a quadratic function?



What about a quadratic function?



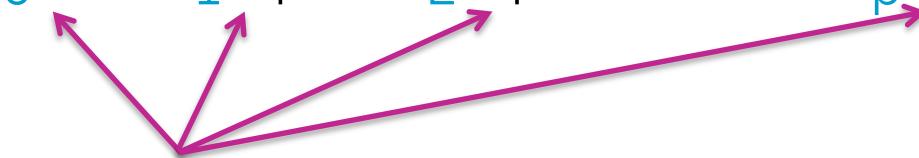
Even higher order polynomial



Polynomial regression

Model:

$$y_i = w_0 + w_1 x_i + w_2 x_i^2 + \dots + w_p x_i^p + \epsilon_i$$



feature 1 = 1 (constant) parameter 1 = w_0

feature 2 = x parameter 2 = w_1

feature 3 = x^2 parameter 3 = w_2

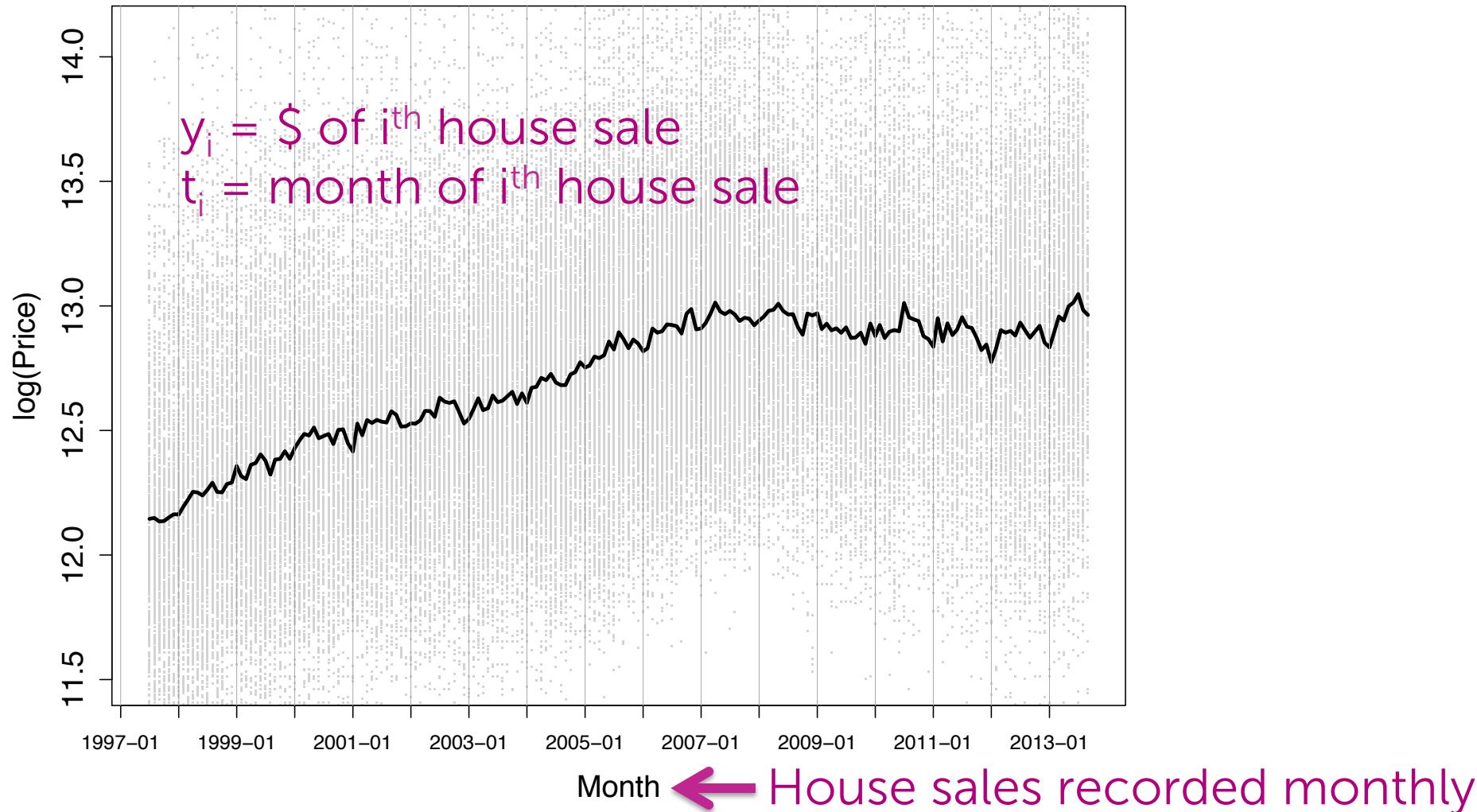
...

...

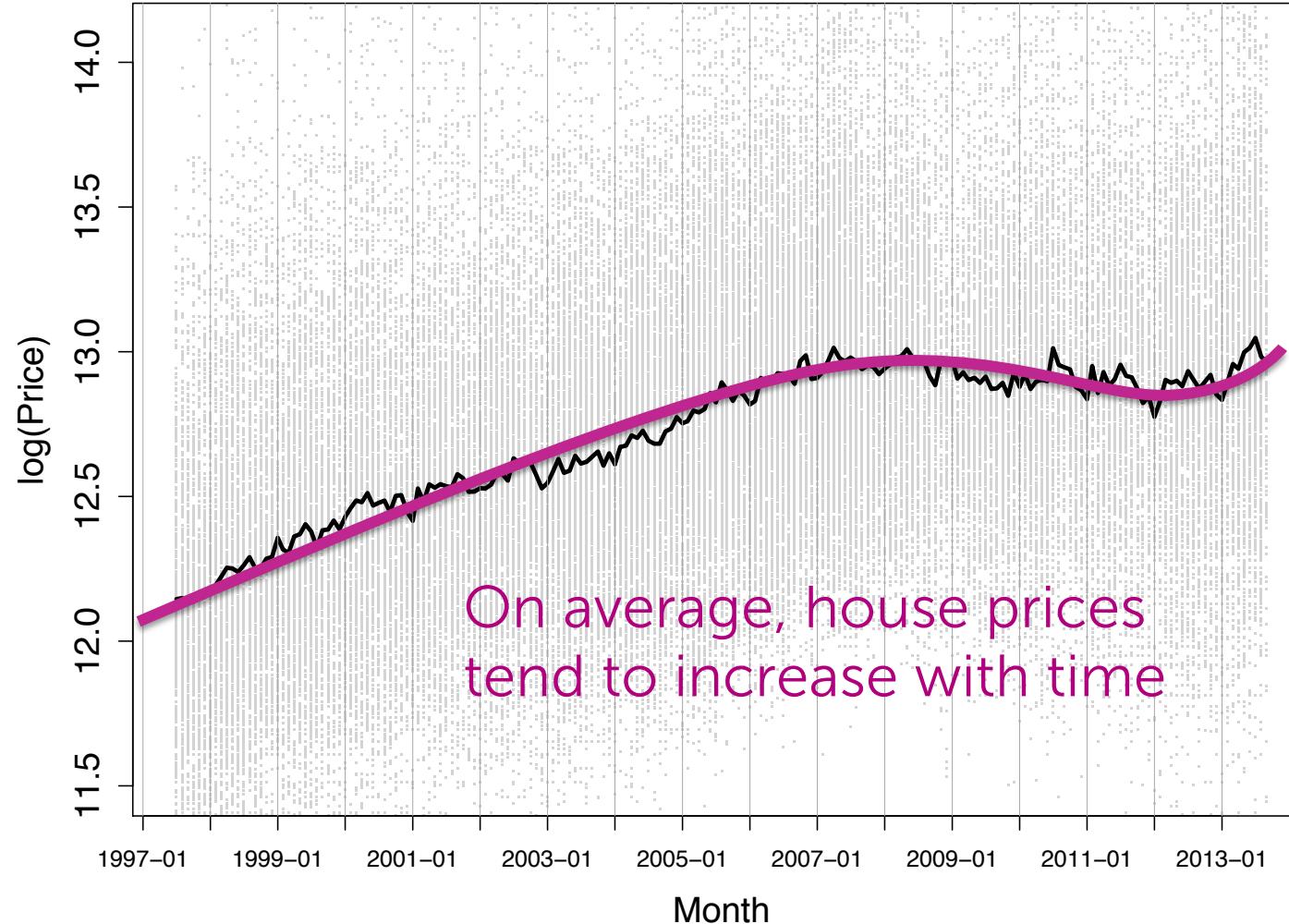
feature $p+1 = x^p$ parameter $p+1 = w_p$

Other functions of one input

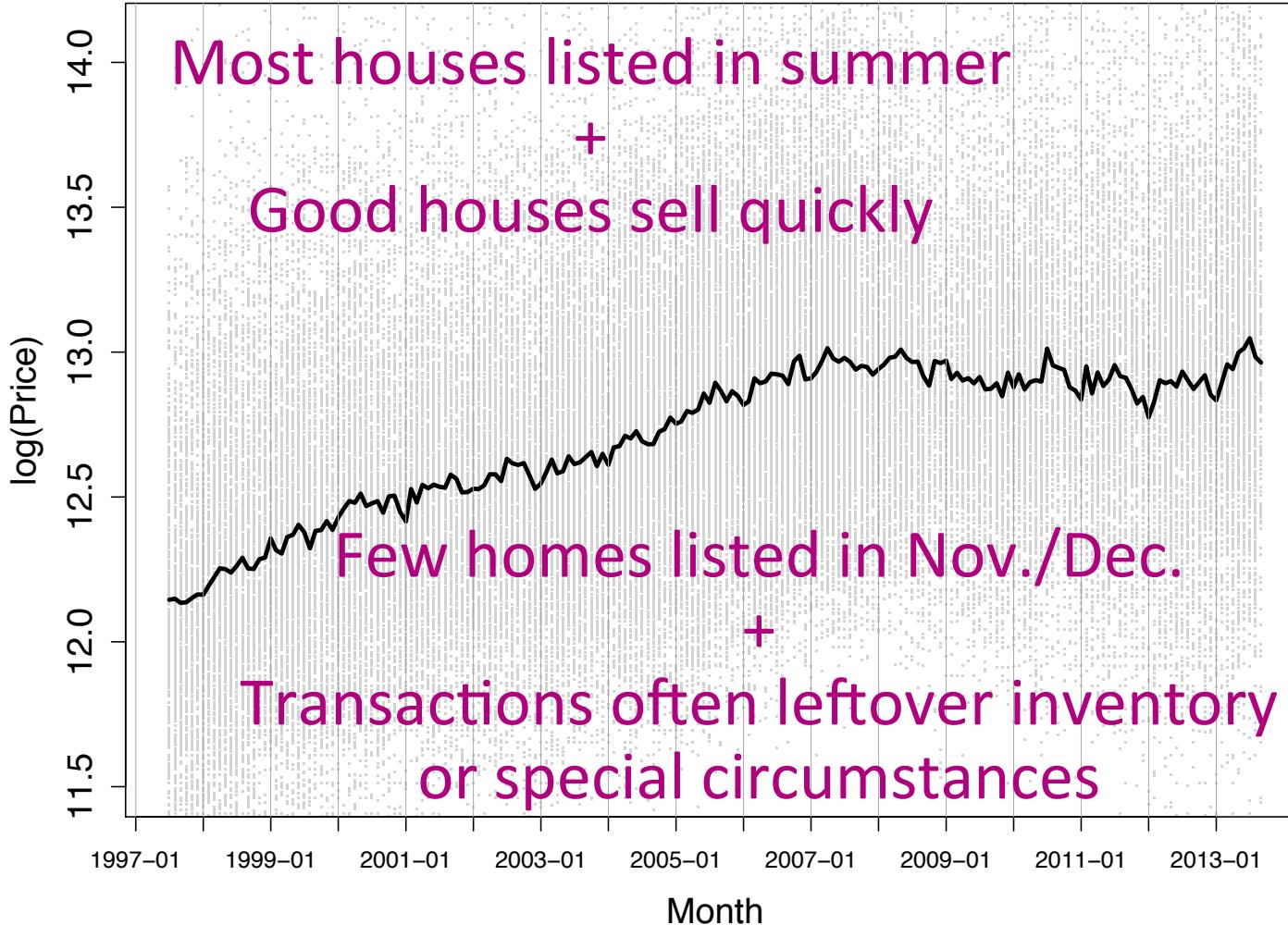
Motivating application: Detrending time series



Trends over time



Seasonality



An example detrending

Model:

$$y_i = w_0 + w_1 t_i + w_2 \sin(2\pi t_i / 12 - \Phi) + \epsilon_i$$

Linear trend

Seasonal component =
Sinusoid with period 12
(resets annually)



Trigonometric identity: $\sin(a-b) = \sin(a)\cos(b) - \cos(a)\sin(b)$

$$\rightarrow \sin(2\pi t_i / 12 - \Phi) = \sin(2\pi t_i / 12)\cos(\Phi) - \cos(2\pi t_i / 12)\sin(\Phi)$$

An example detrending

Equivalently,

$$y_i = w_0 + w_1 t_i + w_2 \sin(2\pi t_i / 12) + w_3 \cos(2\pi t_i / 12) + \epsilon_i$$

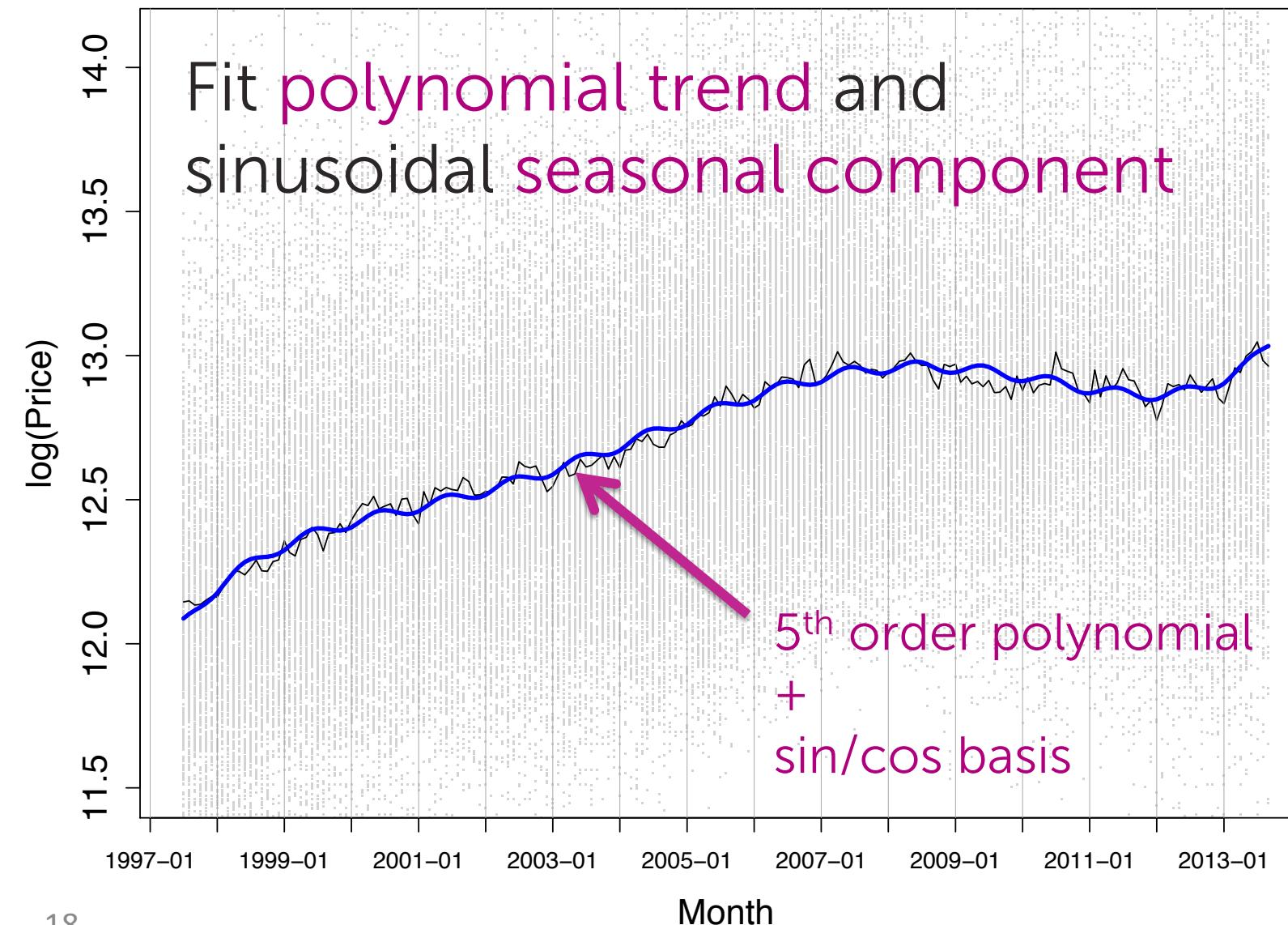
feature 1 = 1 (constant)

feature 2 = t

feature 3 = sin(2πt/12)

feature 4 = cos(2πt/12)

Detrended housing data

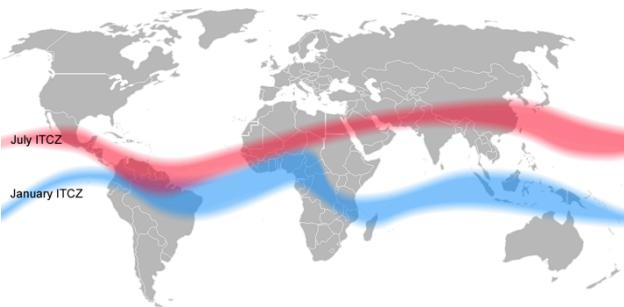


Zoom in...

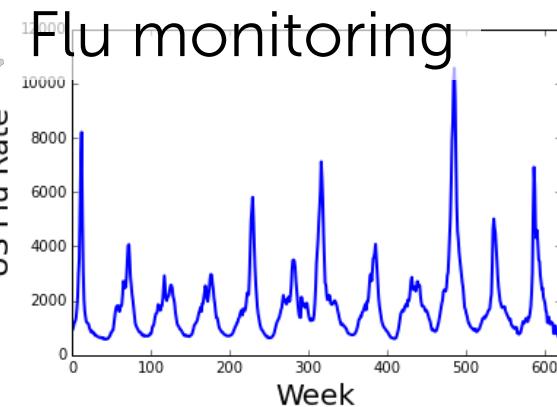
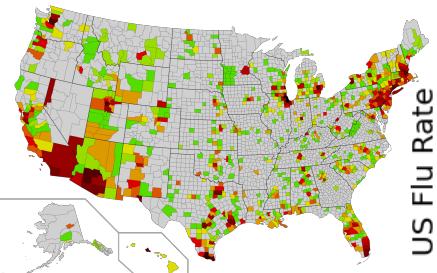
Fit polynomial trend and sinusoidal seasonal component



Other examples of seasonality



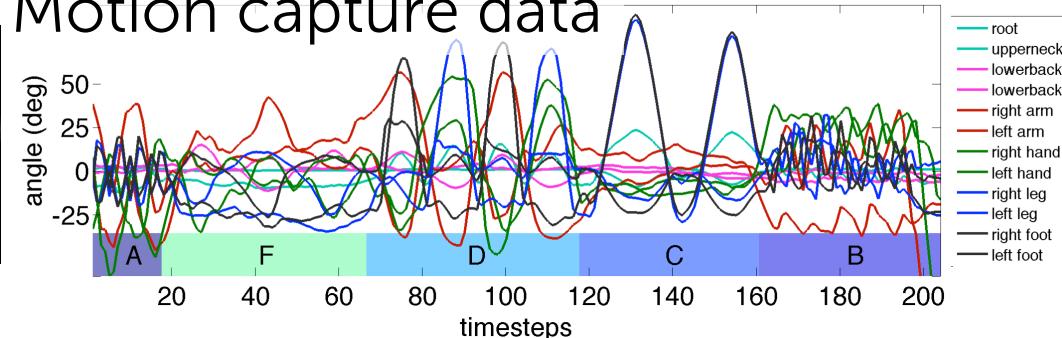
Weather modeling
(e.g., temperature, rainfall)



Demand forecasting
(e.g., jacket purchases)



Motion capture data



More generally...

Generic basis expansion

Model:

$$\begin{aligned}y_i &= w_0 h_0(x_i) + w_1 h_1(x_i) + \dots + w_D h_D(x_i) + \varepsilon_i \\&= \sum_{j=0}^D w_j h_j(x_i) + \varepsilon_i\end{aligned}$$

jth regression coefficient or weight

jth feature

Generic basis expansion

Model:

$$\begin{aligned}y_i &= w_0 h_0(x_i) + w_1 h_1(x_i) + \dots + w_D h_D(x_i) + \varepsilon_i \\&= \sum_{j=0}^D w_j h_j(x_i) + \varepsilon_i\end{aligned}$$

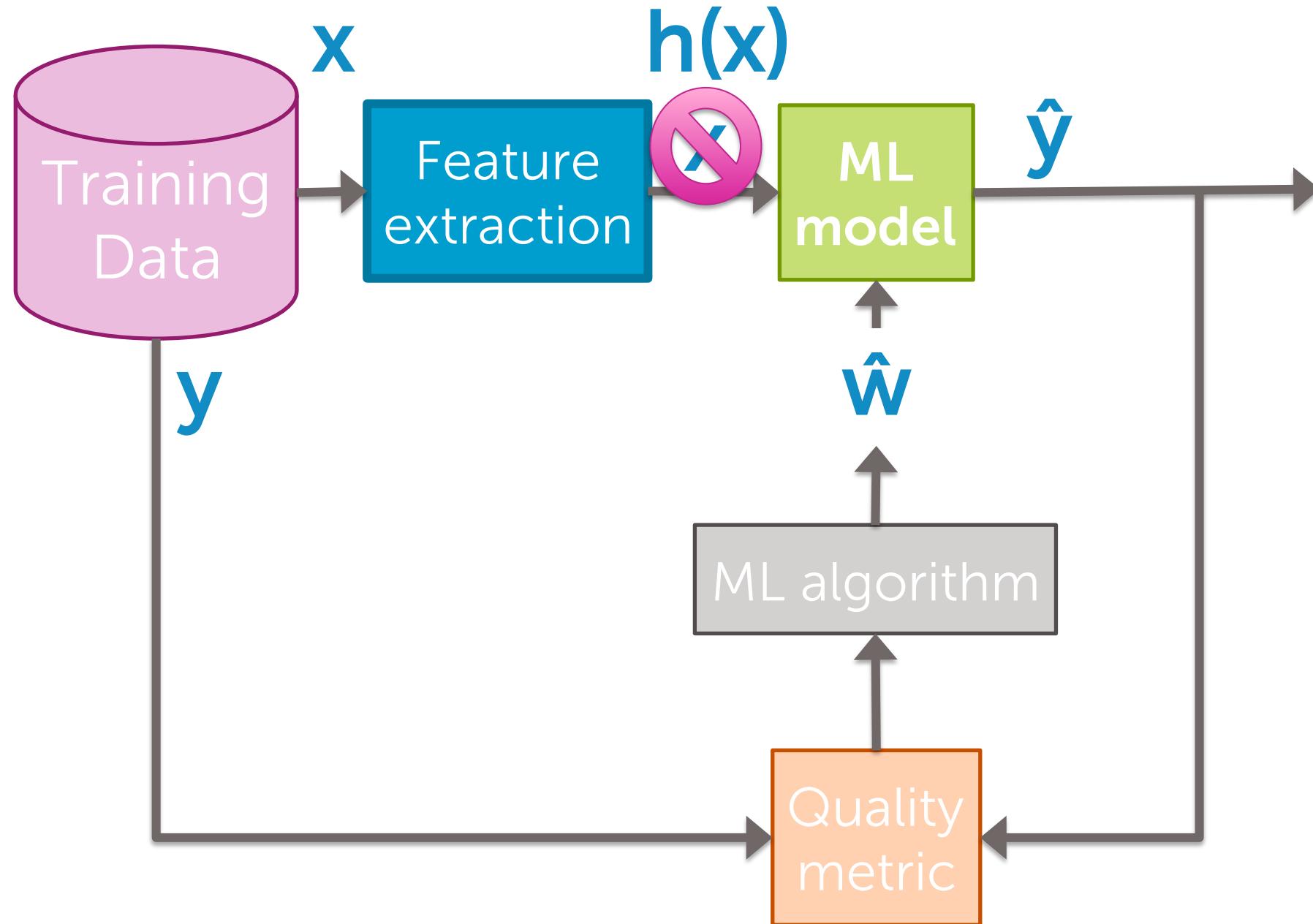
feature 1 = $h_0(x)$... often 1 (constant)

feature 2 = $h_1(x)$... e.g., x

feature 3 = $h_2(x)$... e.g., x^2 or $\sin(2\pi x/12)$

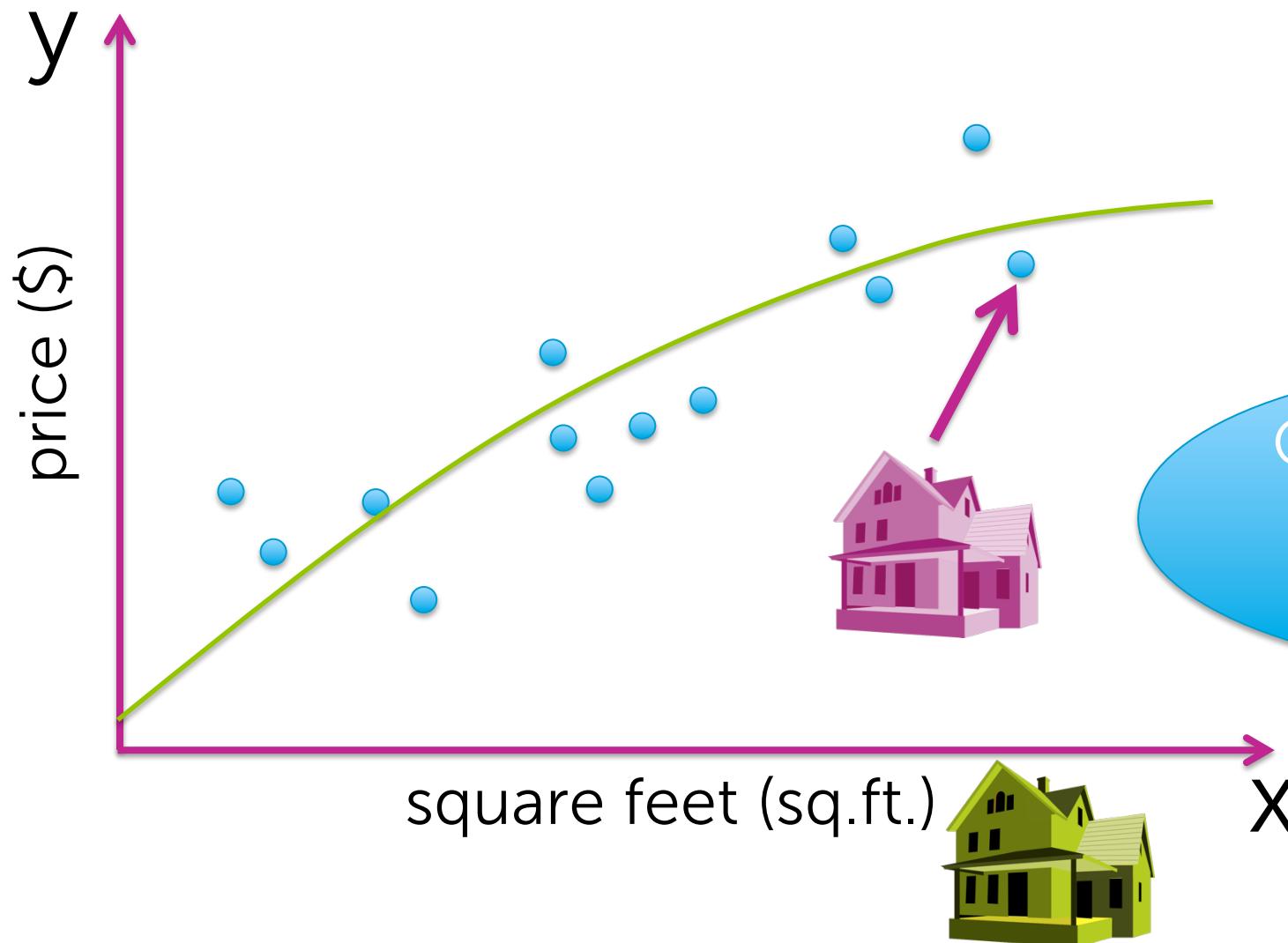
...

feature $D+1 = h_D(x)$... e.g., x^p

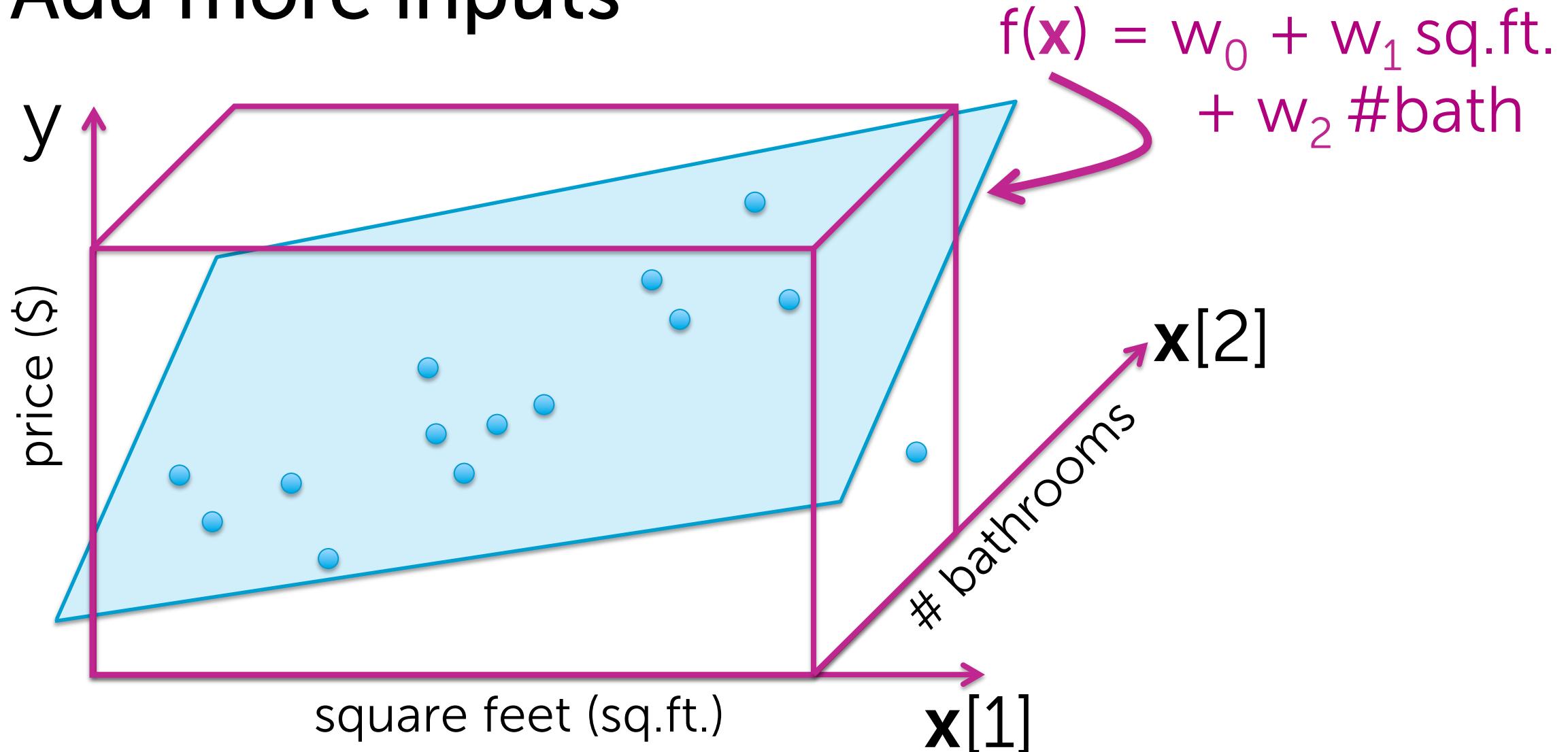


Incorporating multiple inputs

Predictions just based on house size



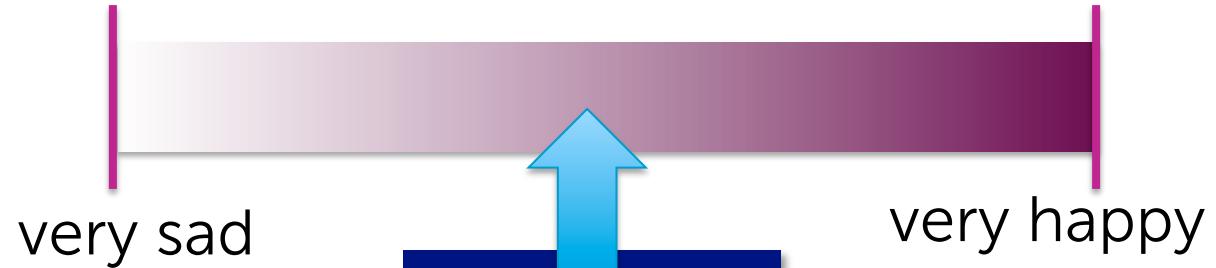
Add more inputs



Many possible inputs

- Square feet
- # bathrooms
- # bedrooms
- Lot size
- Year built
- ...

Reading your mind



Features are
brain region
intensities

General notation

Output: $y \leftarrow$ scalar

Inputs: $\mathbf{x} = (\mathbf{x}[1], \mathbf{x}[2], \dots, \mathbf{x}[d])$

\uparrow
d-dim vector

Notational conventions:

$\mathbf{x}[j] = j^{\text{th}}$ input (scalar)

$h_j(\mathbf{x}) = j^{\text{th}}$ feature (scalar)

$\mathbf{x}_i =$ input of i^{th} data point (vector)

$\mathbf{x}_i[j] = j^{\text{th}}$ input of i^{th} data point (scalar)

Simple hyperplane

Model:

$$y_i = w_0 + w_1 x_i[1] + \dots + w_d x_i[d] + \epsilon_i$$

feature 1 = 1

feature 2 = $x[1]$... e.g., sq. ft.

feature 3 = $x[2]$... e.g., #bath

...

feature $d+1 = x[d]$... e.g., lot size

More generically... D-dimensional curve

Model:

$$\begin{aligned}y_i &= w_0 h_0(\mathbf{x}_i) + w_1 h_1(\mathbf{x}_i) + \dots + w_D h_D(\mathbf{x}_i) + \epsilon_i \\&= \sum_{j=0}^D w_j h_j(\mathbf{x}_i) + \epsilon_i\end{aligned}$$

feature 1 = $h_0(\mathbf{x})$... e.g., 1

feature 2 = $h_1(\mathbf{x})$... e.g., $\mathbf{x}[1]$ = sq. ft.

feature 3 = $h_2(\mathbf{x})$... e.g., $\mathbf{x}[2]$ = #bath
or, $\log(\mathbf{x}[7]) \mathbf{x}[2] = \log(\#bed) \times \#bath$

...

feature D+1 = $h_D(\mathbf{x})$... some other function of $\mathbf{x}[1], \dots, \mathbf{x}[d]$

More on notation

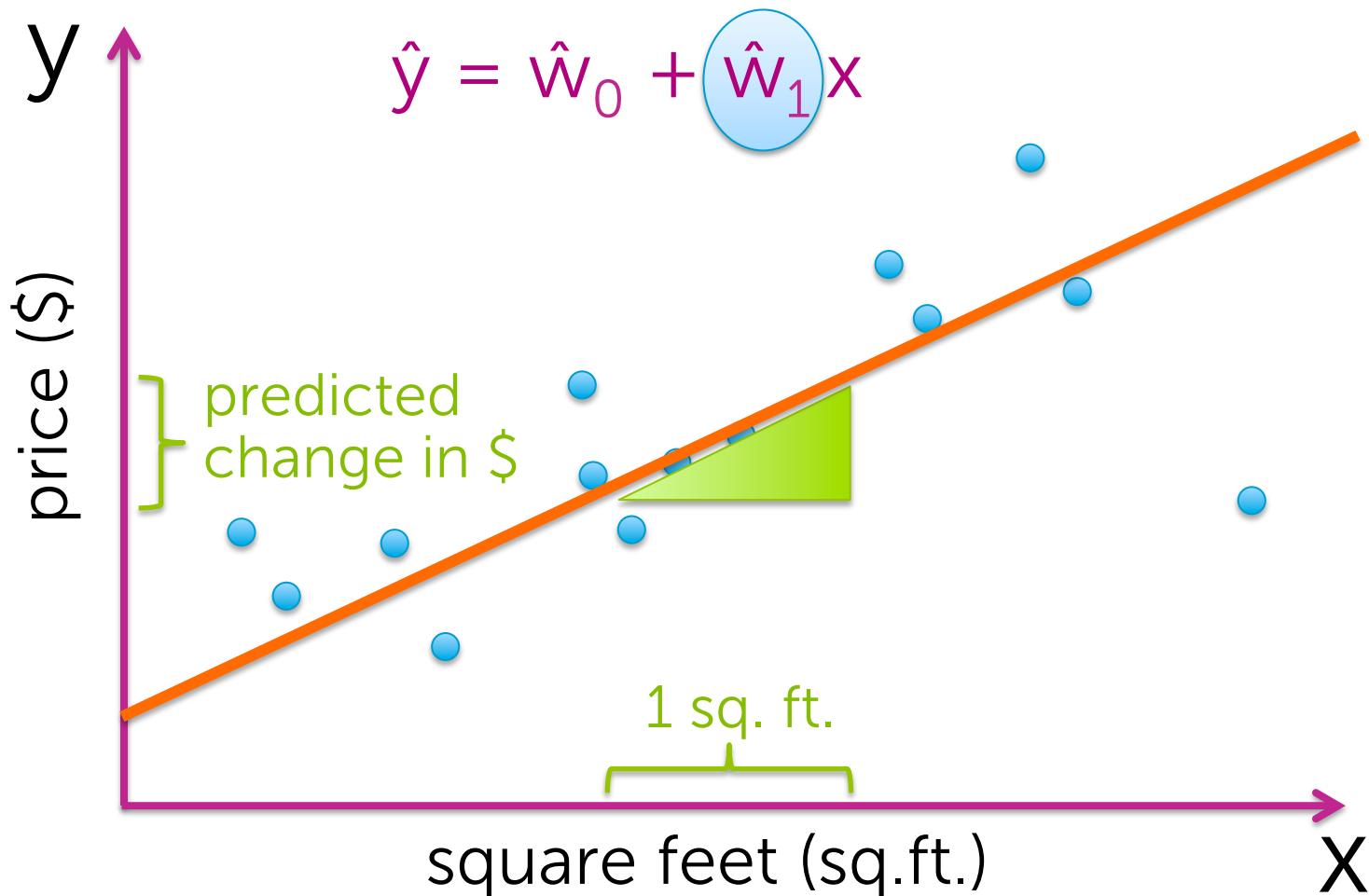
observations (\mathbf{x}_i, y_i) : N

inputs $\mathbf{x}[j]$: d

features $h_j(\mathbf{x})$: D

Interpreting the fitted function

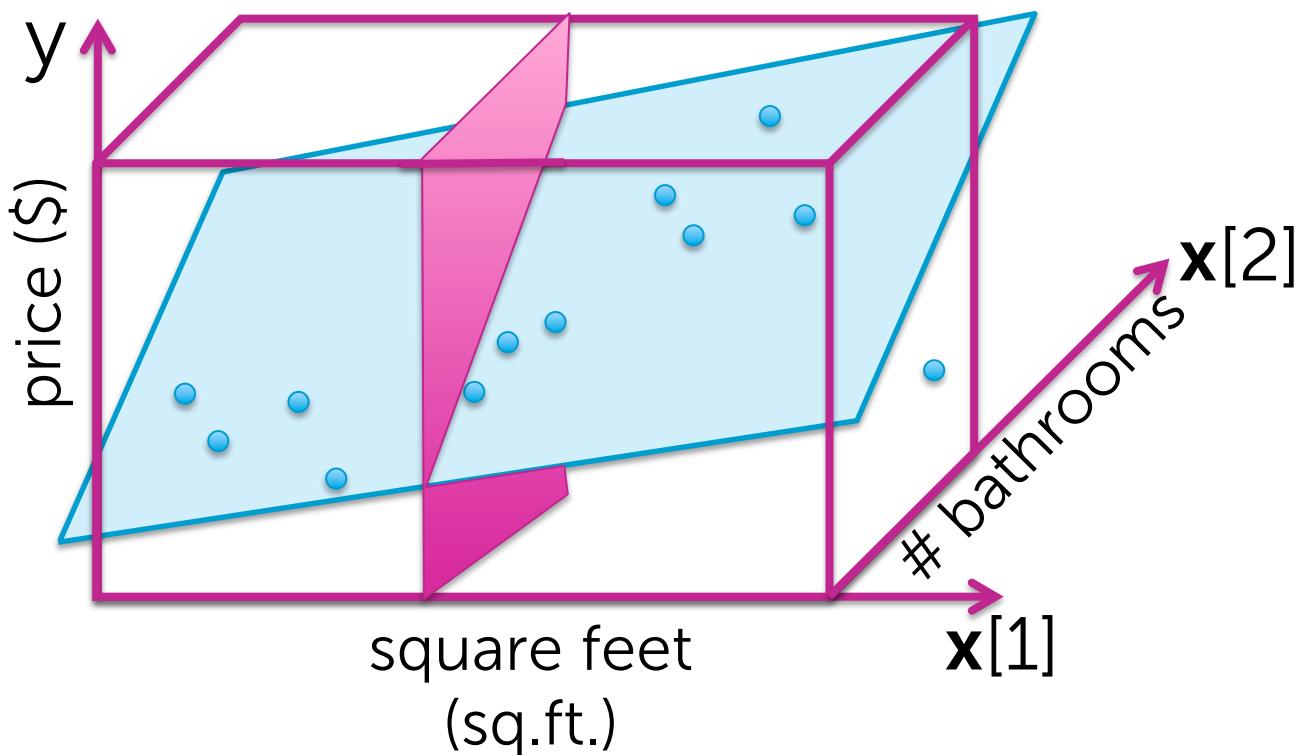
Interpreting the coefficients – Simple linear regression



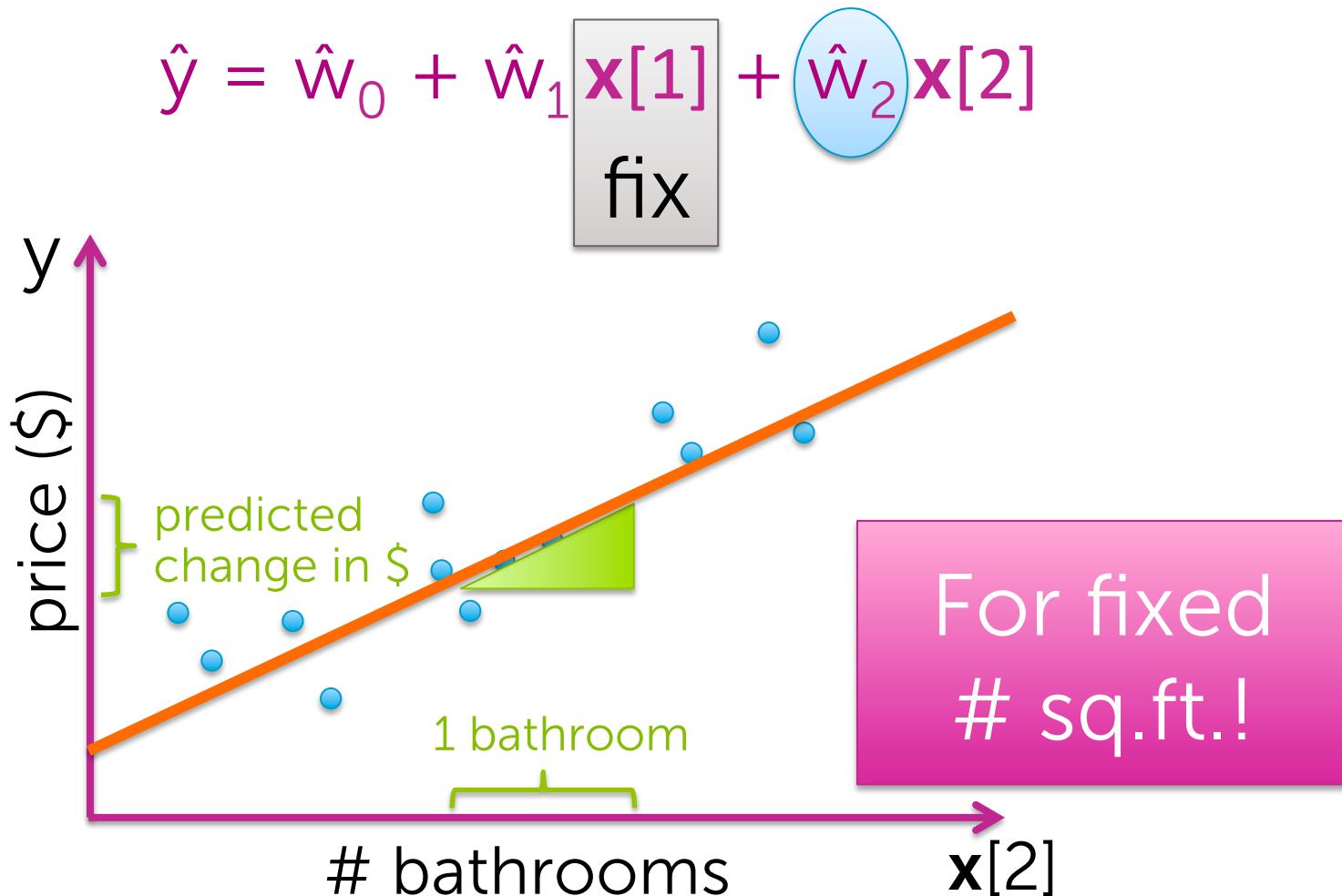
Interpreting the coefficients – Two linear features

$$\hat{y} = \hat{w}_0 + \hat{w}_1 x[1] + \hat{w}_2 x[2]$$

fix



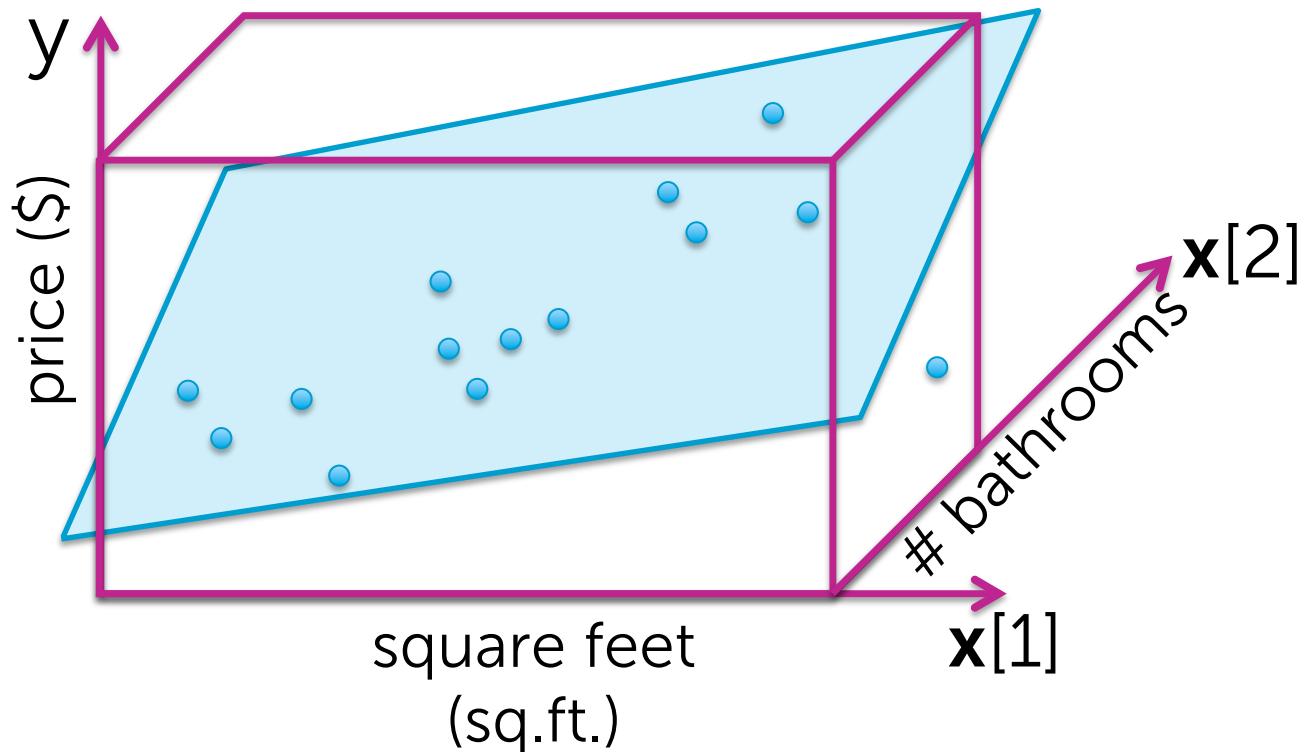
Interpreting the coefficients – Two linear features



Interpreting the coefficients – Multiple linear features

$$\hat{y} = \hat{w}_0 + \hat{w}_1 x[1] + \dots + \hat{w}_j x[j] + \dots + \hat{w}_d x[d]$$

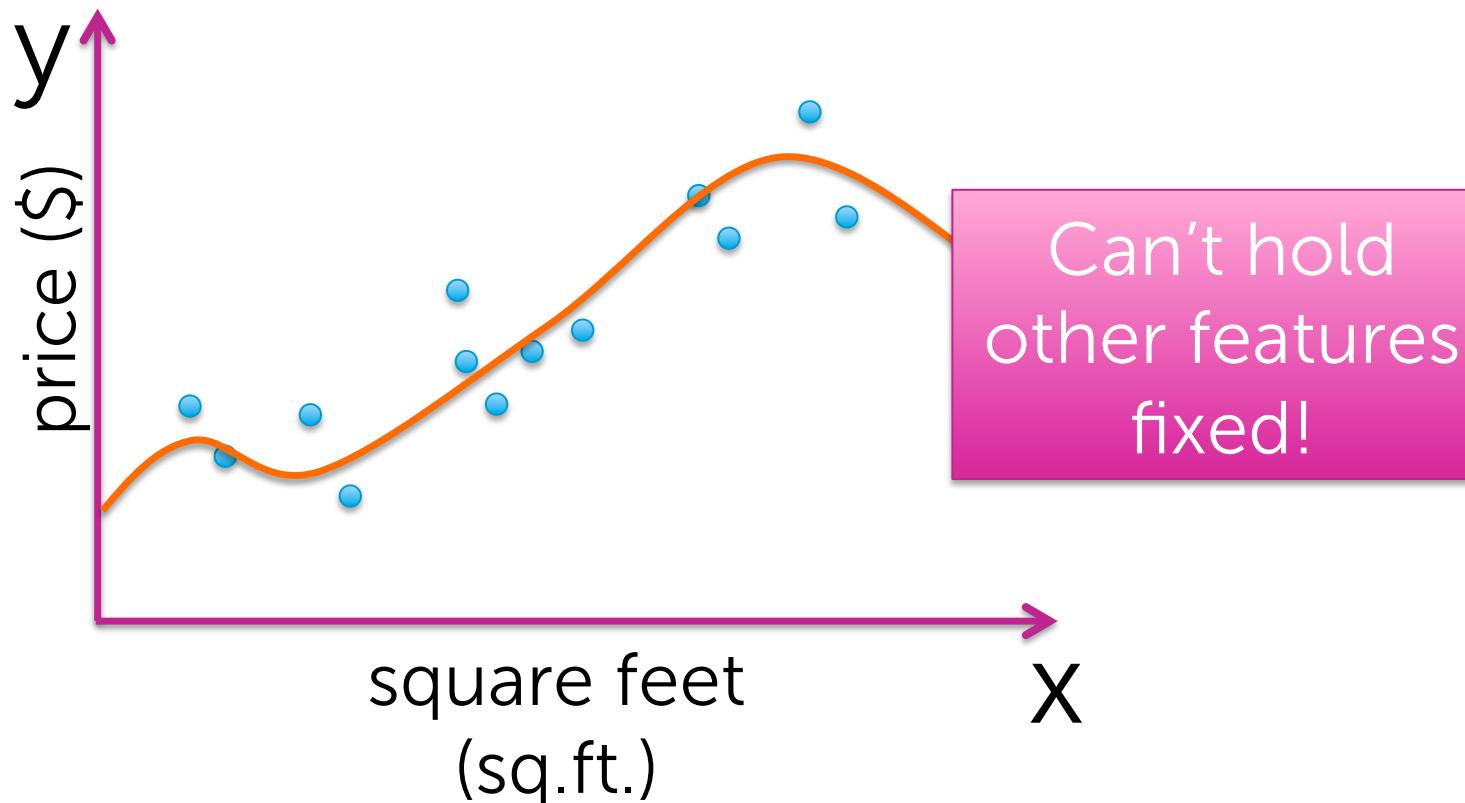
fix fix fix fix



When you think about the interpretation of the coefficient you have to be very careful that the interpretation is in the context of what you have in the model.

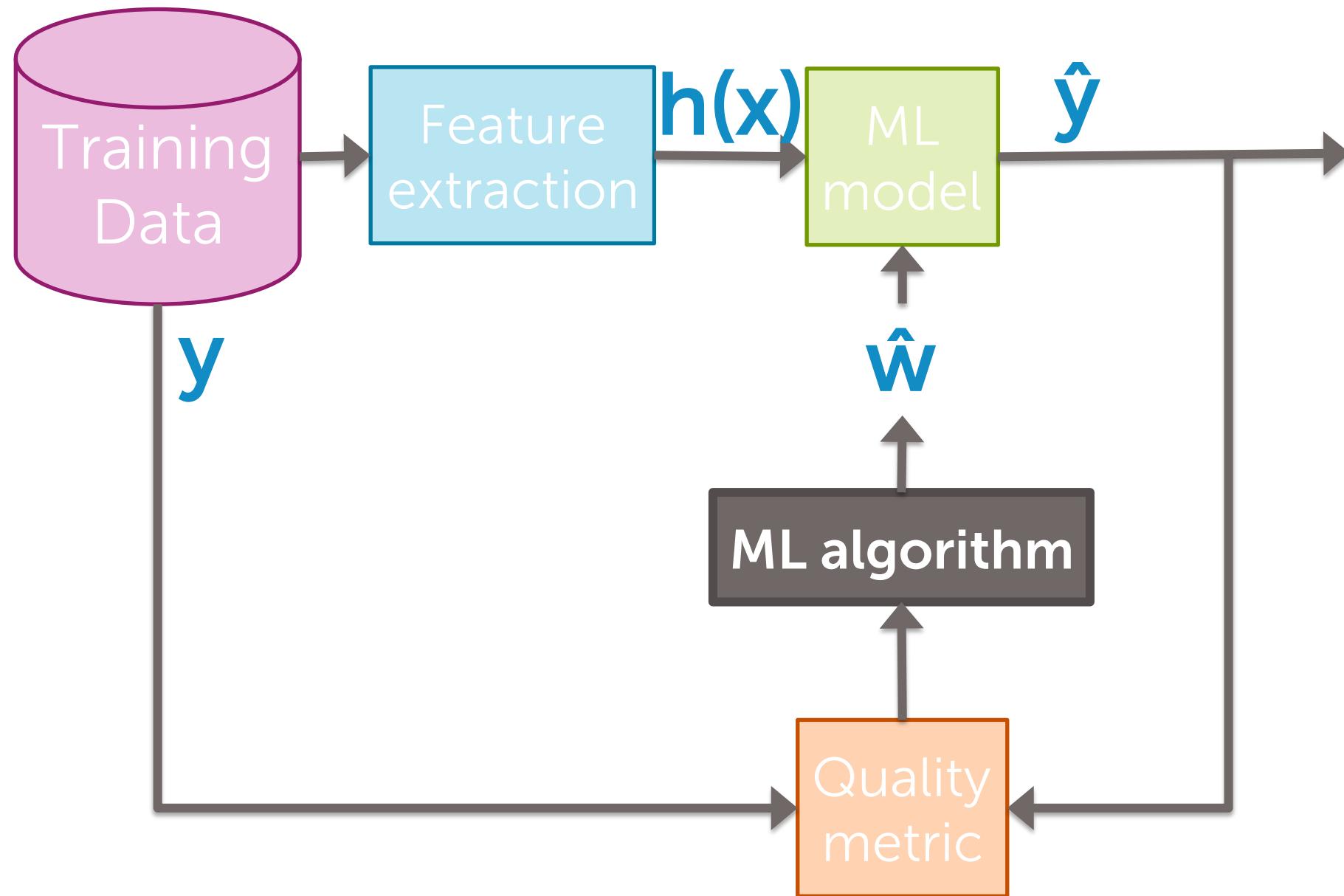
Interpreting the coefficients- Polynomial regression

$$\hat{y} = \hat{w}_0 + \hat{w}_1 x + \dots + \hat{w}_j x^j + \dots + \hat{w}_p x^p$$



If you're ever in a situation where you can't hold all your other features fixed, then you can't think about interpreting the coefficient.

Fitting D-dimensional curves



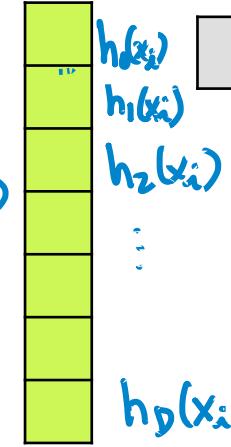
Step 1:

Rewrite the regression model

Rewrite in matrix notation

For observation i

$$y_i = \sum_{j=0}^D w_j h_j(x_i) + \epsilon_i$$

y_i =  $\underbrace{w^\top}_{\text{scalar}}$  $= w_0 h_0(x_i) + w_1 h_1(x_i) + \dots + w_D h_D(x_i) + \epsilon_i$

$$= w^\top h(x_i) + \epsilon_i$$

$$\begin{aligned} & h^\top(x_i) \\ & = [h_0(x_i) \ h_1(x_i) \ \dots \ h_D(x_i)] \\ & = h_0(x_i) w_0 + h_1(x_i) w_1 + \dots + h_D(x_i) w_D + \epsilon_i \end{aligned}$$

$$w = \begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_D \end{bmatrix}$$

Rewrite in matrix notation

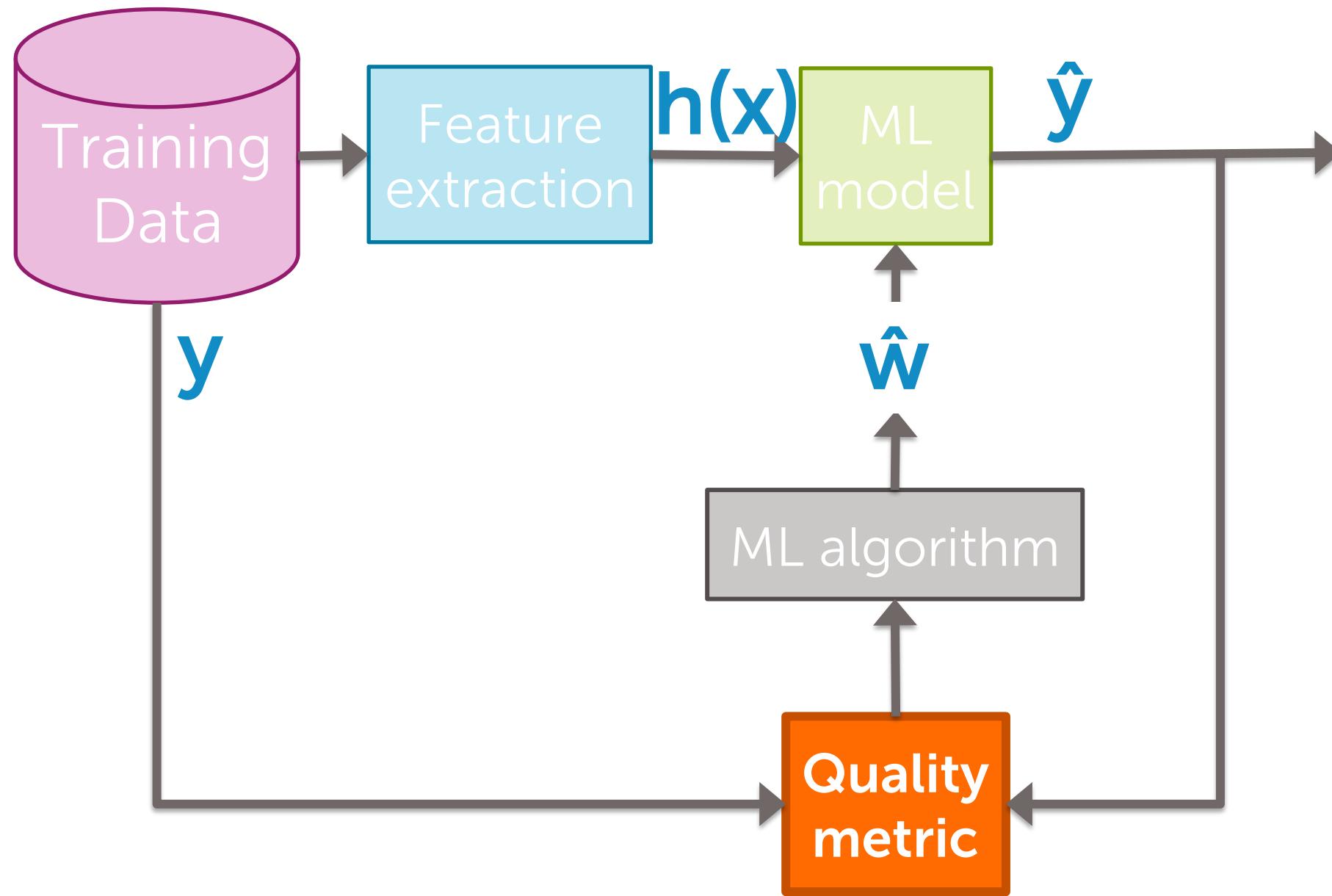
For all observations together

$$\begin{matrix} \mathbf{y} \\ \mathbf{y}_1 \\ \mathbf{y}_2 \\ \mathbf{y}_3 \\ \vdots \\ \vdots \\ \mathbf{y}_N \end{matrix} = \mathbf{H} \begin{matrix} \mathbf{w} \\ w_0 \\ w_1 \\ w_2 \\ \vdots \\ w_D \end{matrix} + \begin{matrix} \mathbf{\epsilon} \\ \epsilon_1 \\ \epsilon_2 \\ \epsilon_3 \\ \vdots \\ \vdots \\ \epsilon_N \end{matrix}$$

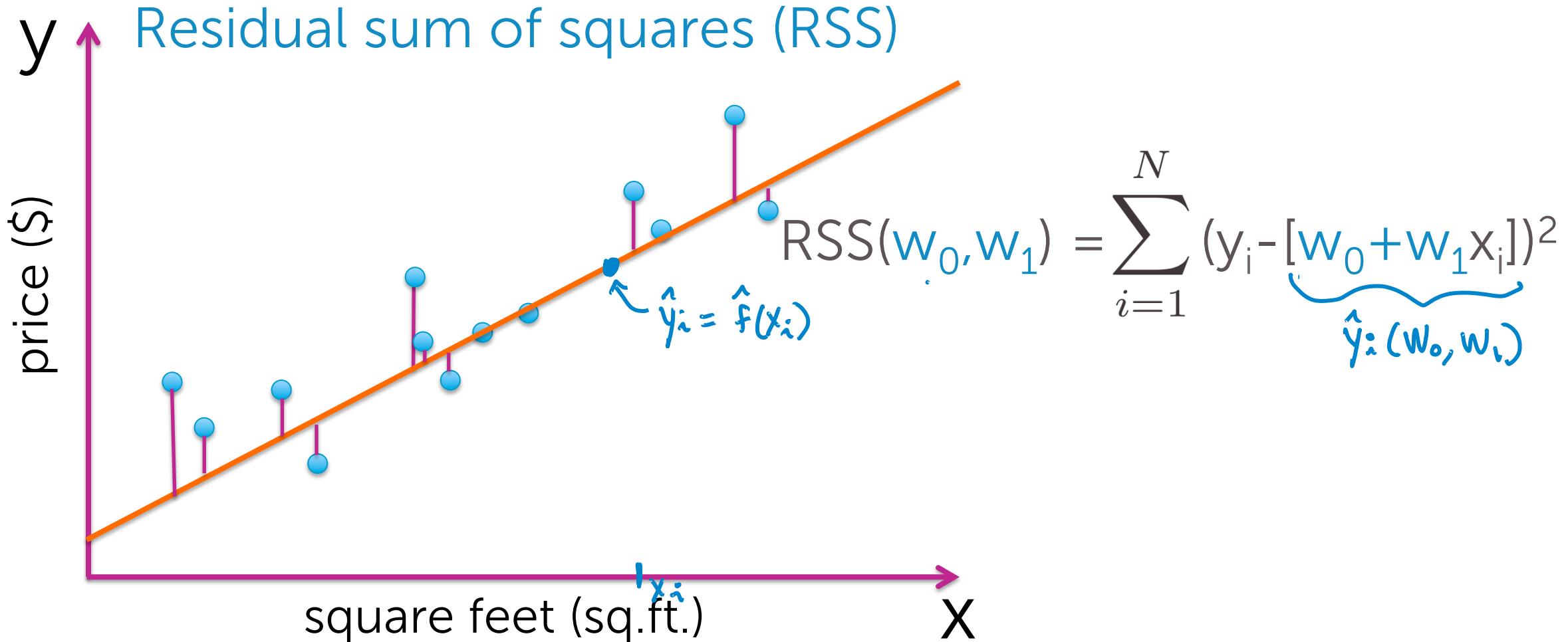
$\Rightarrow \boxed{\mathbf{y} = \mathbf{H}\mathbf{w} + \mathbf{\epsilon}}$

A handwritten note above the matrix \mathbf{H} indicates $h^T(x_i)$, pointing to the first column of the matrix.

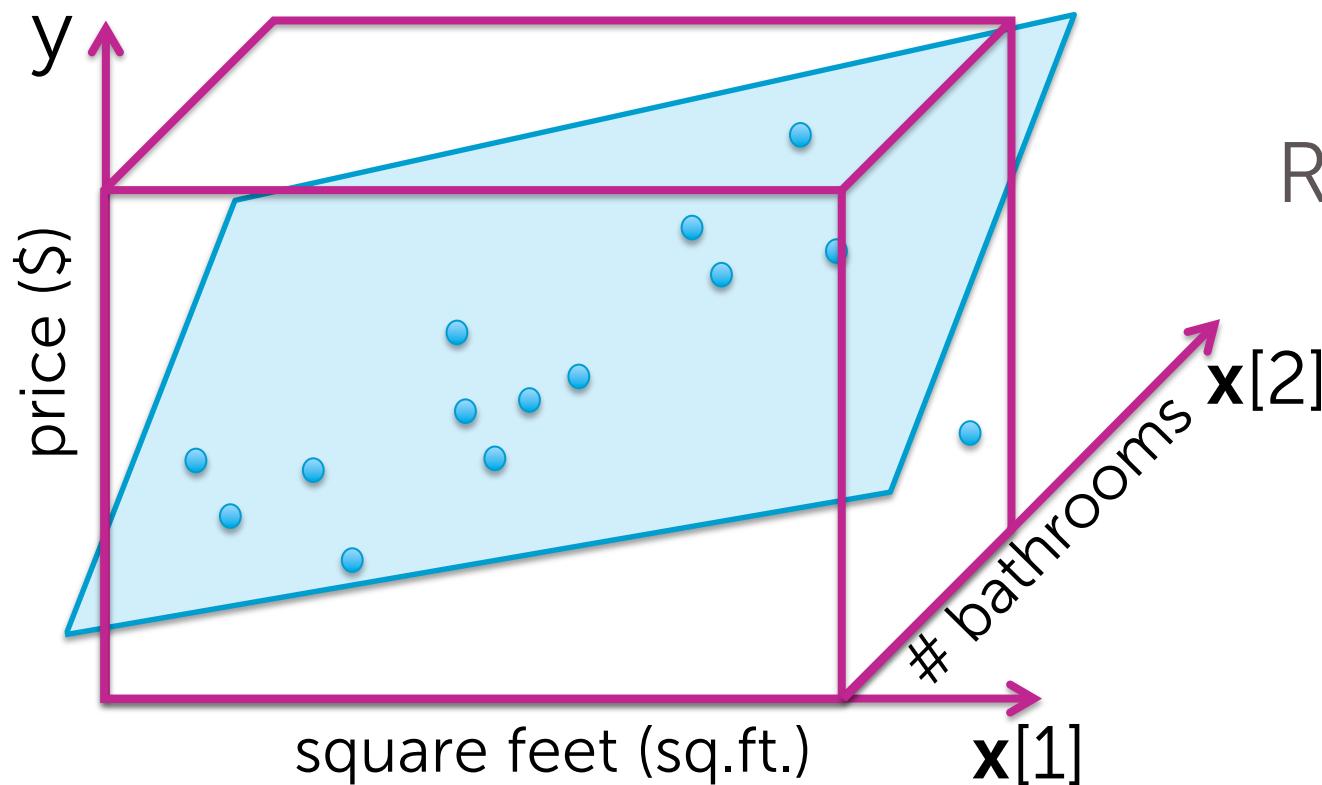
Step 2: Compute the cost



"Cost" of using a given line



RSS for multiple regression



$$\text{RSS}(\underline{\mathbf{w}}) = \sum_{i=1}^N (y_i - \hat{y}_i(\underline{\mathbf{w}}))^2$$
$$\hat{y}_i = \begin{bmatrix} h_0(x_i) & h_1(x_i) & \dots & h_D(x_i) \end{bmatrix} \underline{\mathbf{w}}$$
$$\underline{\mathbf{w}} = \begin{bmatrix} w_0 \\ w_1 \\ w_2 \\ \vdots \\ w_D \end{bmatrix}$$

RSS in matrix notation

$$\text{RSS}(\mathbf{w}) = \sum_{i=1}^N (y_i - h(\mathbf{x}_i)^T \mathbf{w})^2$$

$$= (\mathbf{y} - \mathbf{H}\mathbf{w})^T (\mathbf{y} - \mathbf{H}\mathbf{w})$$

Why? (part 1)

$$\hat{\mathbf{y}} = \begin{bmatrix} \hat{y}_1 \\ \hat{y}_2 \\ \vdots \\ \hat{y}_N \end{bmatrix}$$

$$= \mathbf{H} = \begin{bmatrix} & & & & \\ & & & & \\ & & & & \\ & & & & \\ & & & & \end{bmatrix}$$

$$\mathbf{w} = \begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_D \end{bmatrix}$$

$$\hat{\mathbf{y}} = \mathbf{H}\mathbf{w}$$

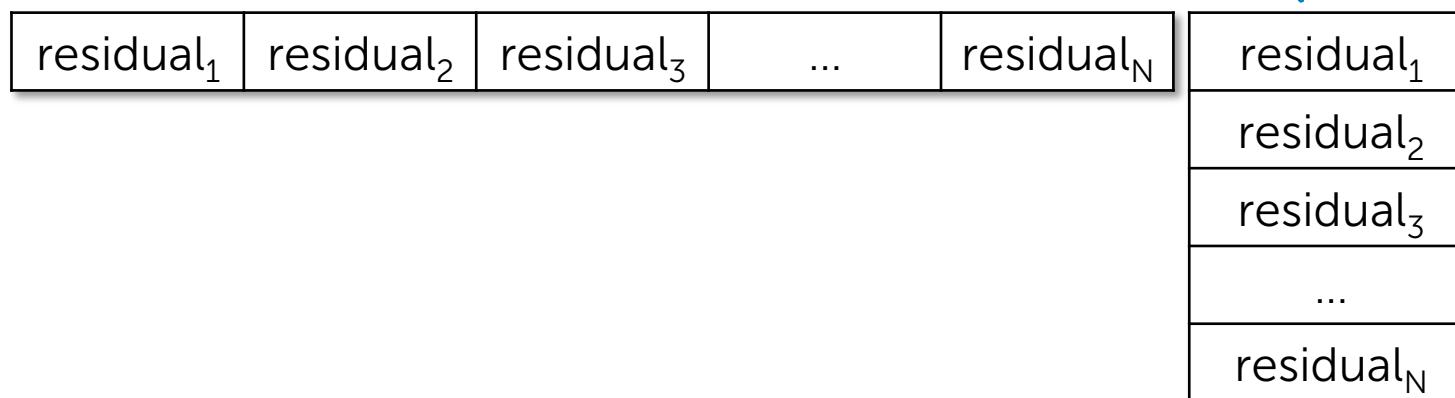
$$(\mathbf{y} - \mathbf{H}\mathbf{w}) = (\mathbf{y} - \hat{\mathbf{y}}) = \begin{bmatrix} \text{residual}_1 \\ \text{residual}_2 \\ \vdots \\ \text{residual}_N \end{bmatrix}$$

$$\text{residual}_i = y_i - \hat{y}_i$$

RSS in matrix notation

$$\text{RSS}(\mathbf{w}) = \sum_{i=1}^N (y_i - h(\mathbf{x}_i)^\top \mathbf{w})^2$$
$$= (\mathbf{y} - \mathbf{H}\mathbf{w})^\top (\mathbf{y} - \mathbf{H}\mathbf{w})$$

Why? (part 2)



$$\begin{aligned} & (\text{residual}_1^2 + \text{residual}_2^2 + \dots + \text{residual}_N^2) \\ &= \sum_{i=1}^N \text{residual}_i^2 \\ &\triangleq \text{RSS}(\mathbf{w}) \end{aligned}$$

Step 3:

Take the gradient

Gradient of RSS

$$\nabla_{\mathbf{w}} \text{RSS}(\mathbf{w}) = \nabla [(\mathbf{y} - \mathbf{H}\mathbf{w})^\top (\mathbf{y} - \mathbf{H}\mathbf{w})]$$
$$= -2\mathbf{H}^\top (\mathbf{y} - \mathbf{H}\mathbf{w})$$

Why? By analogy to 1D case:

$$\frac{d}{dw} (y - hw)(y - hw) = \frac{d}{dw} (y - hw)^2 = 2 \cdot (y - hw)' (-h)$$
$$= -2h(y - hw)$$

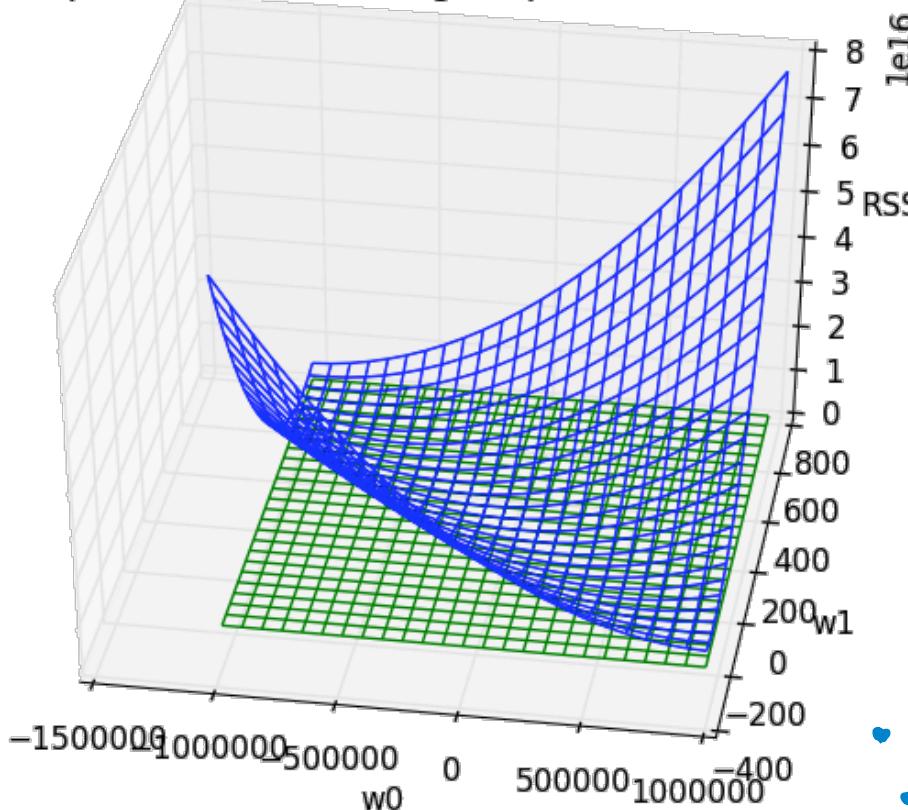
(y - hw) is a scalar

Step 4, Approach 1:

Set the gradient = 0

Closed-form solution

3D plot of RSS with tangent plane at minimum



$$\begin{matrix} \bullet & A^{-1}A = I \\ \bullet & IV = V \\ \bullet & IV = V \end{matrix}$$

$$\nabla \text{RSS}(\mathbf{w}) = -2\mathbf{H}^T(\mathbf{y} - \mathbf{H}\mathbf{w}) = 0$$

Solve for \mathbf{w} :

$$-2\cancel{\mathbf{H}^T}\mathbf{y} + \cancel{2}\mathbf{H}^T\mathbf{H}\hat{\mathbf{w}} = 0$$

$$\mathbf{H}^T\mathbf{H}\hat{\mathbf{w}} = \mathbf{H}^T\mathbf{y}$$

$$\underbrace{(\mathbf{H}^T\mathbf{H})^{-1}}_{I} \mathbf{H}^T\hat{\mathbf{w}} = (\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T\mathbf{y}$$

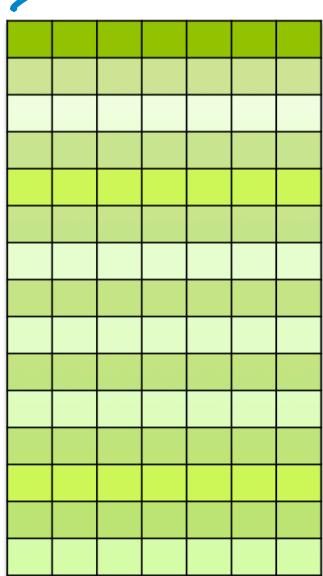
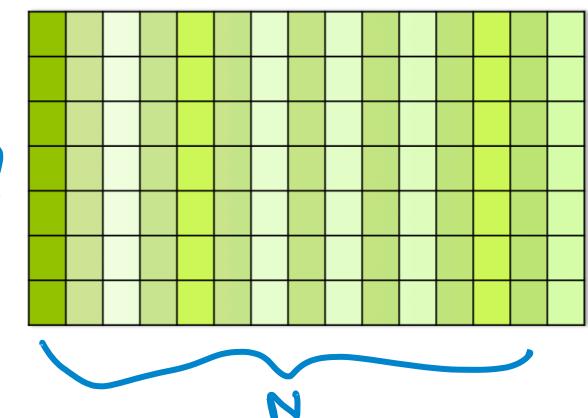
$$\hat{\mathbf{w}} = (\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T\mathbf{y}$$

Closed-form solution

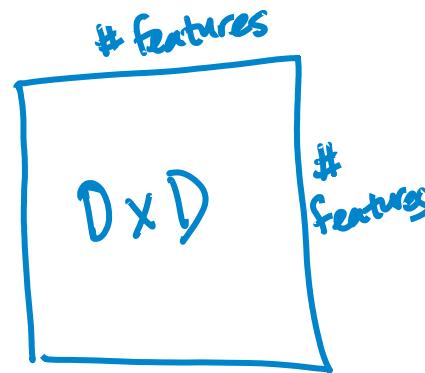
$$\hat{\mathbf{w}} = (\mathbf{H}^T \mathbf{H})^{-1} \mathbf{H}^T \mathbf{y}$$



features = D



obs = N



really,
of linearly
ind. observations

Invertible if:

In most cases is $N > D$

In most case invertible if $N > D$

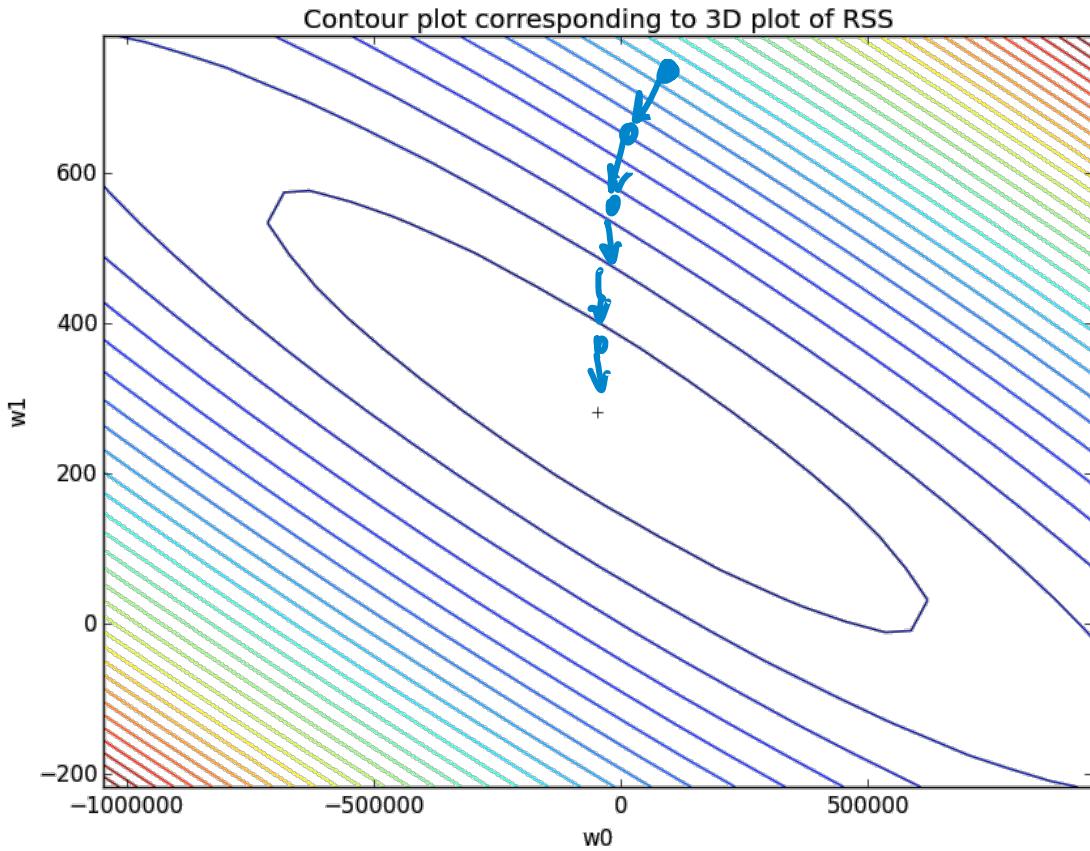
$N = \#$ linearly independent observations

Complexity of inverse:

$O(D^3)$

Step 4, Approach 2: Gradient descent

Gradient descent



while not converged

$$\mathbf{w}^{(t+1)} \leftarrow \mathbf{w}^{(t)} - \eta \nabla \text{RSS}(\mathbf{w}^{(t)})$$
$$= \mathbf{w}^{(t)} + 2\eta \mathbf{H}^T (\mathbf{y} - \mathbf{H}\mathbf{w}^{(t)})$$
$$\hat{\mathbf{y}}(\mathbf{w}^{(t)})$$

Feature-by-feature update

$$\text{RSS}(\mathbf{w}) = \sum_{i=1}^N (y_i - h(\mathbf{x}_i)^T \mathbf{w})^2$$
$$= \sum_{i=1}^N (y_i - w_0 h_0(x_i) - w_1 h_1(x_i) - \dots - w_D h_D(x_i))^2$$

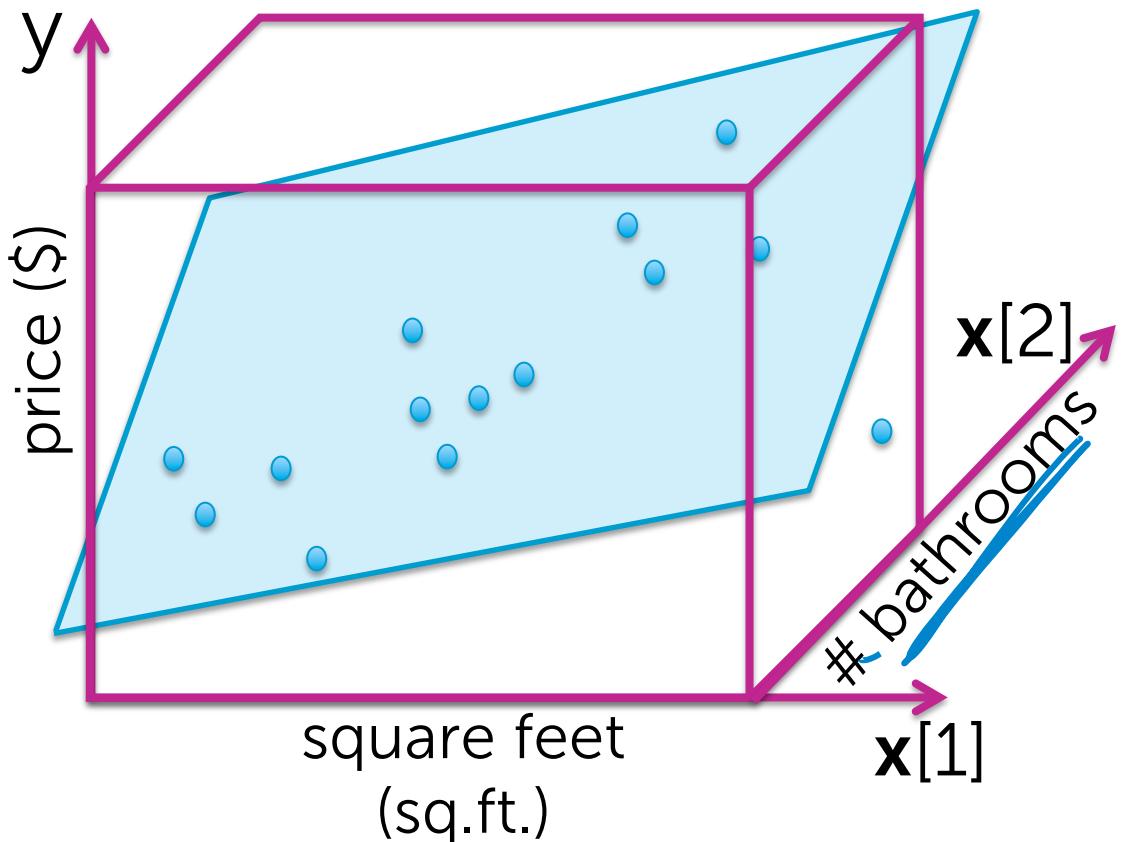
Partial with respect to w_j .

$$\begin{aligned} & \sum_{i=1}^N 2(y_i - w_0 h_0(x_i) - w_1 h_1(x_i) - \dots - w_D h_D(x_i)) \\ & \quad \cdot (-\underline{h_j(x_i)}) \\ &= -2 \sum_{i=1}^N h_j(x_i) (y_i - h(\mathbf{x}_i)^T \mathbf{w}) \end{aligned}$$

Update to j^{th} feature weight:

$$w_j^{(t+1)} \leftarrow w_j^{(t)} - \eta \left(-2 \sum_{i=1}^N h_j(x_i) (y_i - \underbrace{h^T(\mathbf{x}_i) \mathbf{w}^{(t)}}_{\hat{y}_i(\mathbf{w}^{(t)})}) \right)$$

Interpreting elementwise

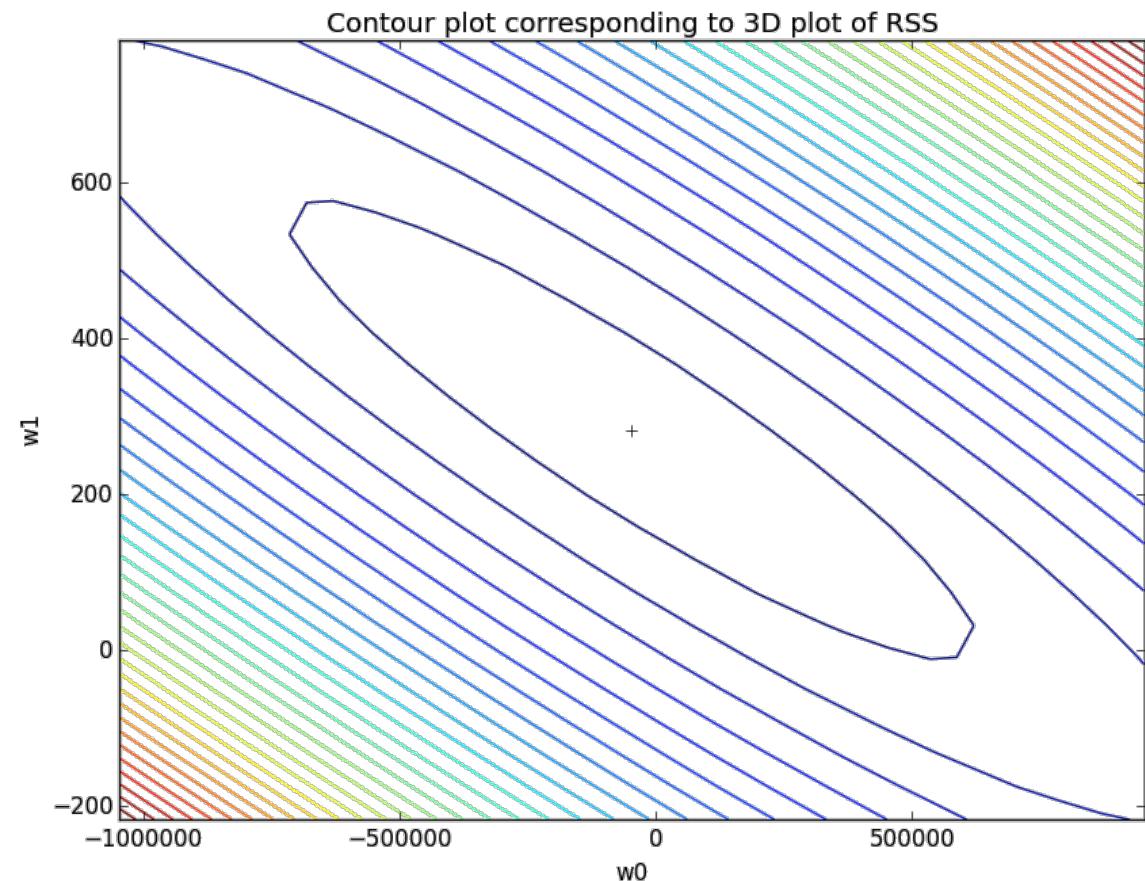


Update to j^{th} feature weight:

$$w_j^{(t+1)} \leftarrow w_j^{(t)} + 2n \sum_{i=1}^N h_j(x_i)(y_i - \hat{y}_i(w^{(t)}))$$

If underestimating impact of #bath ($\hat{w}_j^{(t)}$ is too small)
then $(y_i - \hat{y}_i(w^{(t)}))$ on average
weighted by #bath will be positive
 $\Rightarrow w_j^{(t+1)} > w_j^{(t)}$ (increase)

Summary of gradient descent for multiple regression



```
init  $\mathbf{w}^{(1)} = \mathbf{0}$  (or randomly, or smartly),  $t = 1$ 
while  $\|\nabla \text{RSS}(\mathbf{w}^{(t)})\| > \epsilon$  tolerance
    for  $j = 0, \dots, D$ 
        partial[j] =  $-2 \sum_{i=1}^N h_j(\mathbf{x}_i)(y_i - \hat{y}_i(\mathbf{w}^{(t)}))$ 
         $\mathbf{w}_j^{(t+1)} \leftarrow \mathbf{w}_j^{(t)} - \eta \text{partial}[j]$ 
    t  $\leftarrow t + 1$ 
```

An extremely useful algorithm

Summary for multiple linear regression

What you can do now...

- Describe polynomial regression
- Detrend a time series using trend and seasonal components
- Write a regression model using multiple inputs or features thereof
- Cast both polynomial regression and regression with multiple inputs as regression with multiple features
- Calculate a goodness-of-fit metric (e.g., RSS)
- Estimate model parameters of a general multiple regression model to minimize RSS:
 - In closed form
 - Using an iterative gradient descent algorithm
- Interpret the coefficients of a non-featurized multiple regression fit
- Exploit the estimated model to form predictions
- Explain applications of multiple regression beyond house price modeling