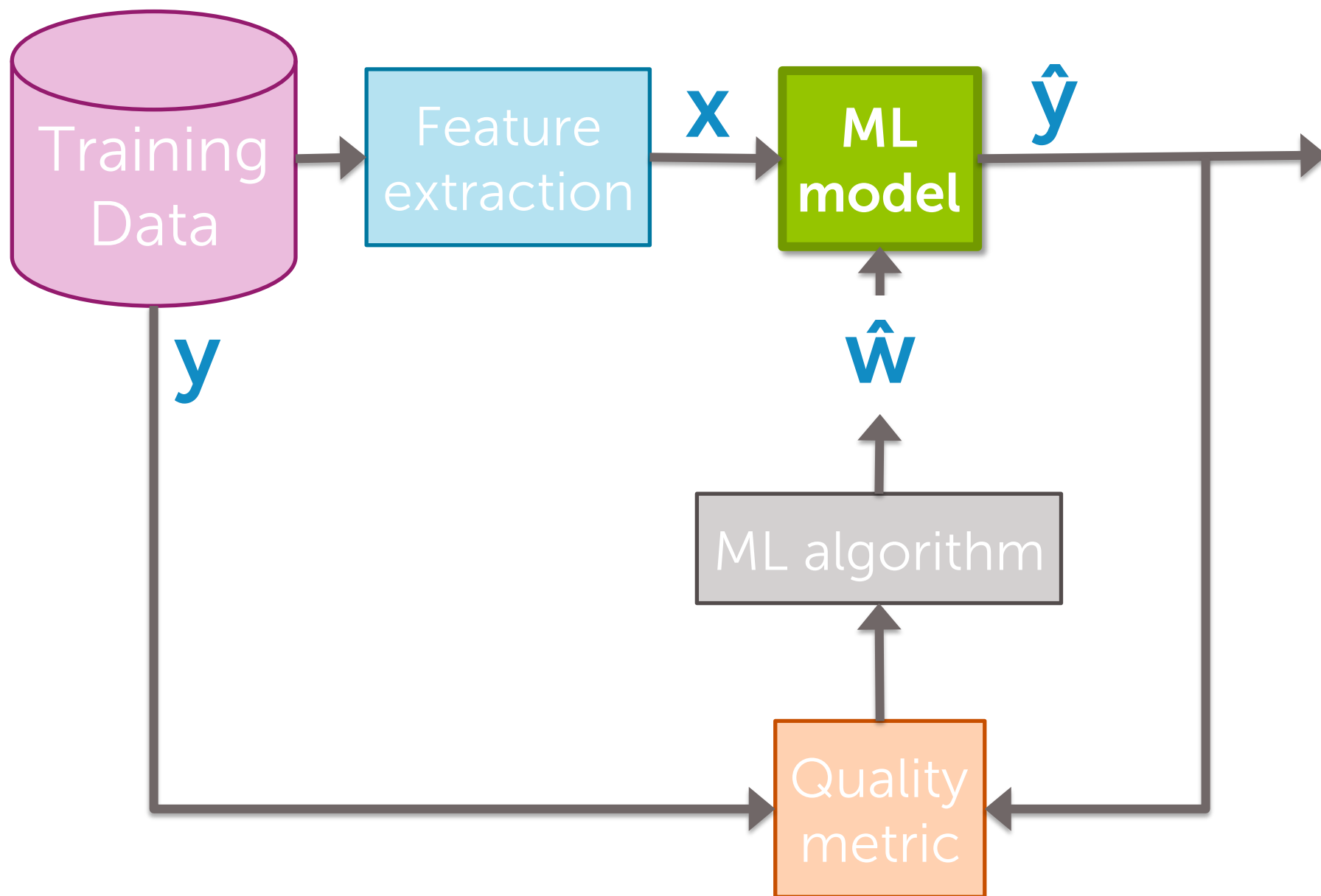


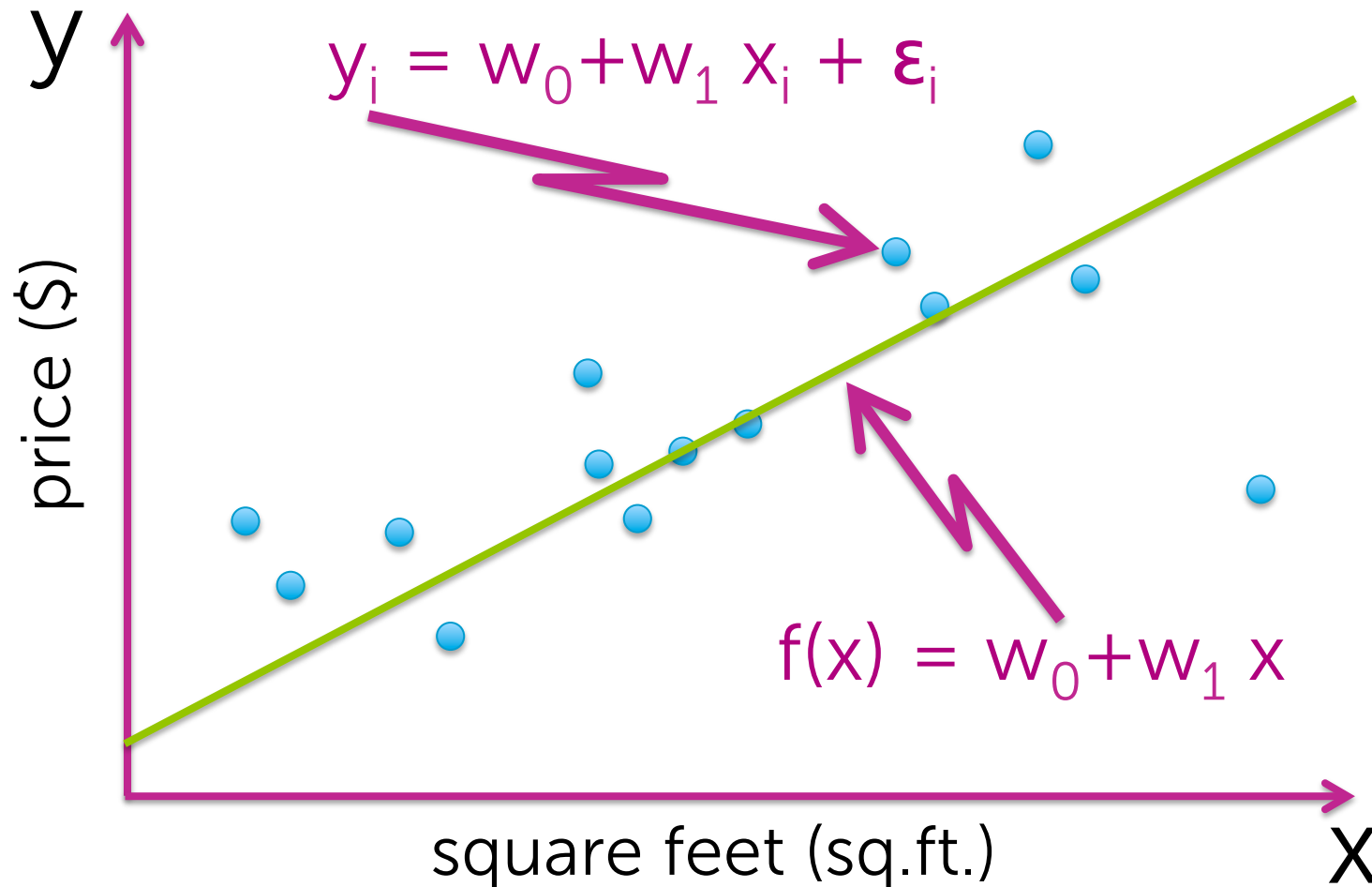
Multiple Regression:

Linear regression with multiple features

Emily Fox & Carlos Guestrin
Machine Learning Specialization
University of Washington



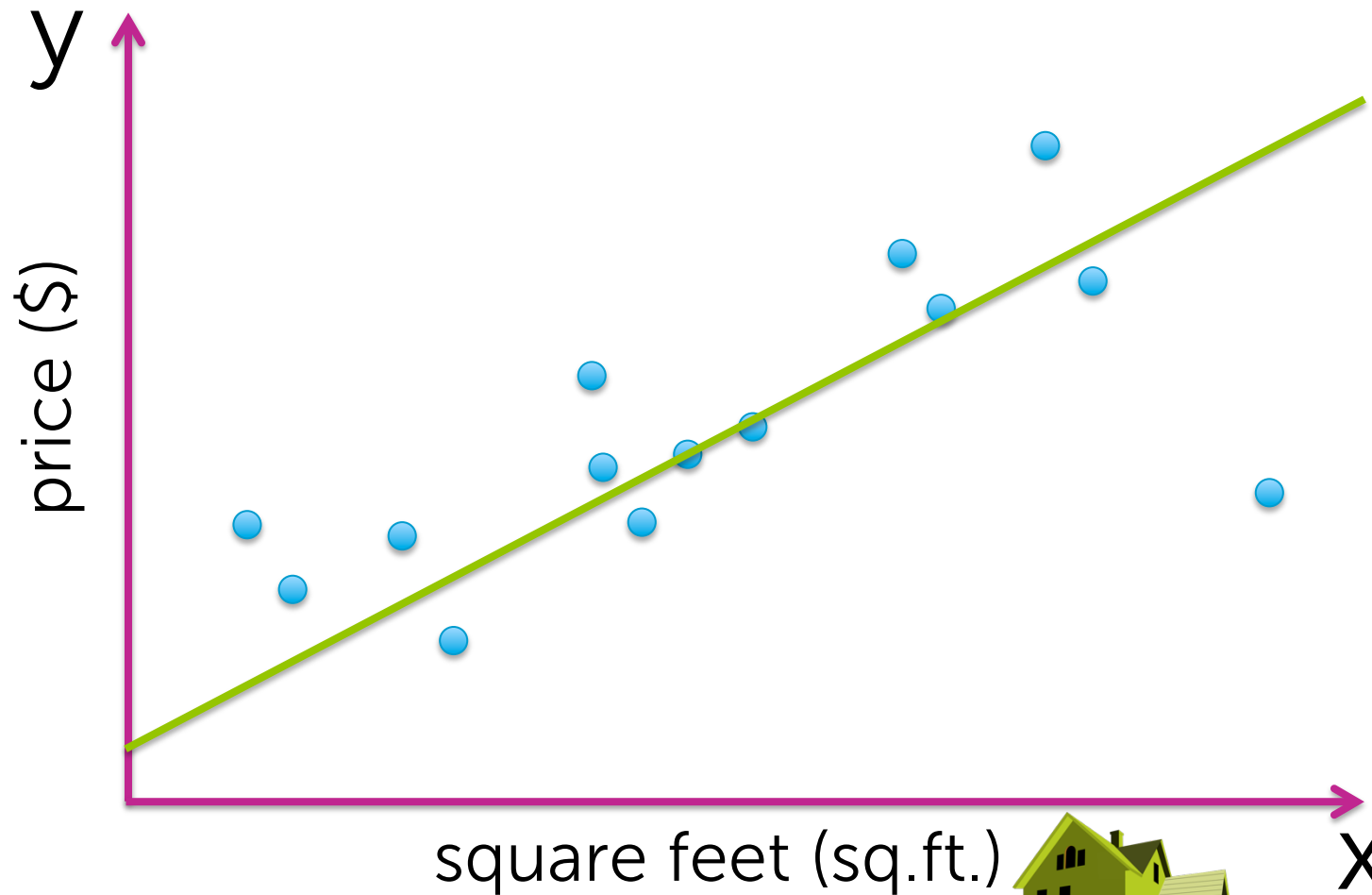
Simple linear regression model



More complex functions of a single input

Polynomial regression

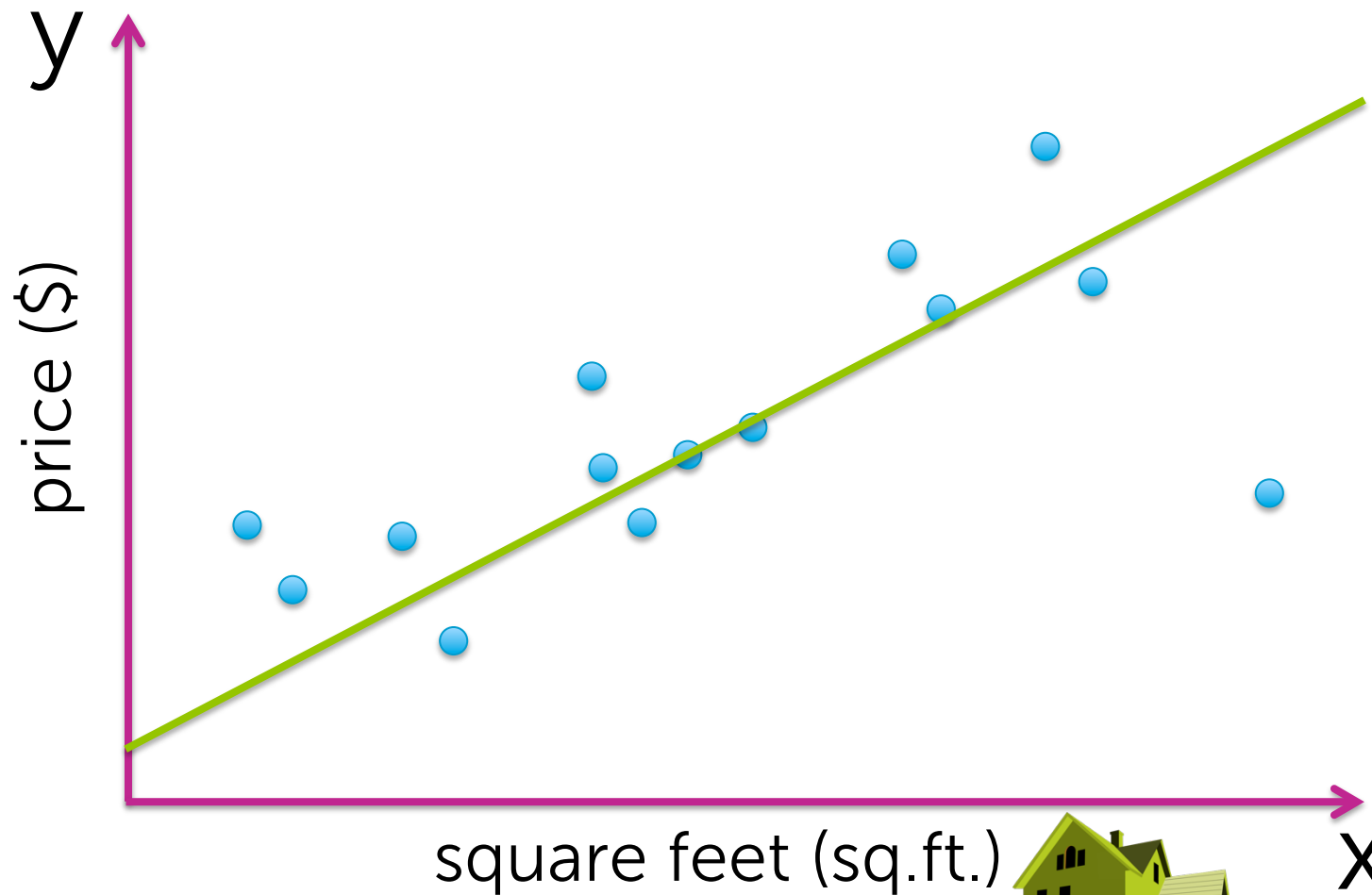
Fit data with a line or ... ?



You show
your friend
your analysis

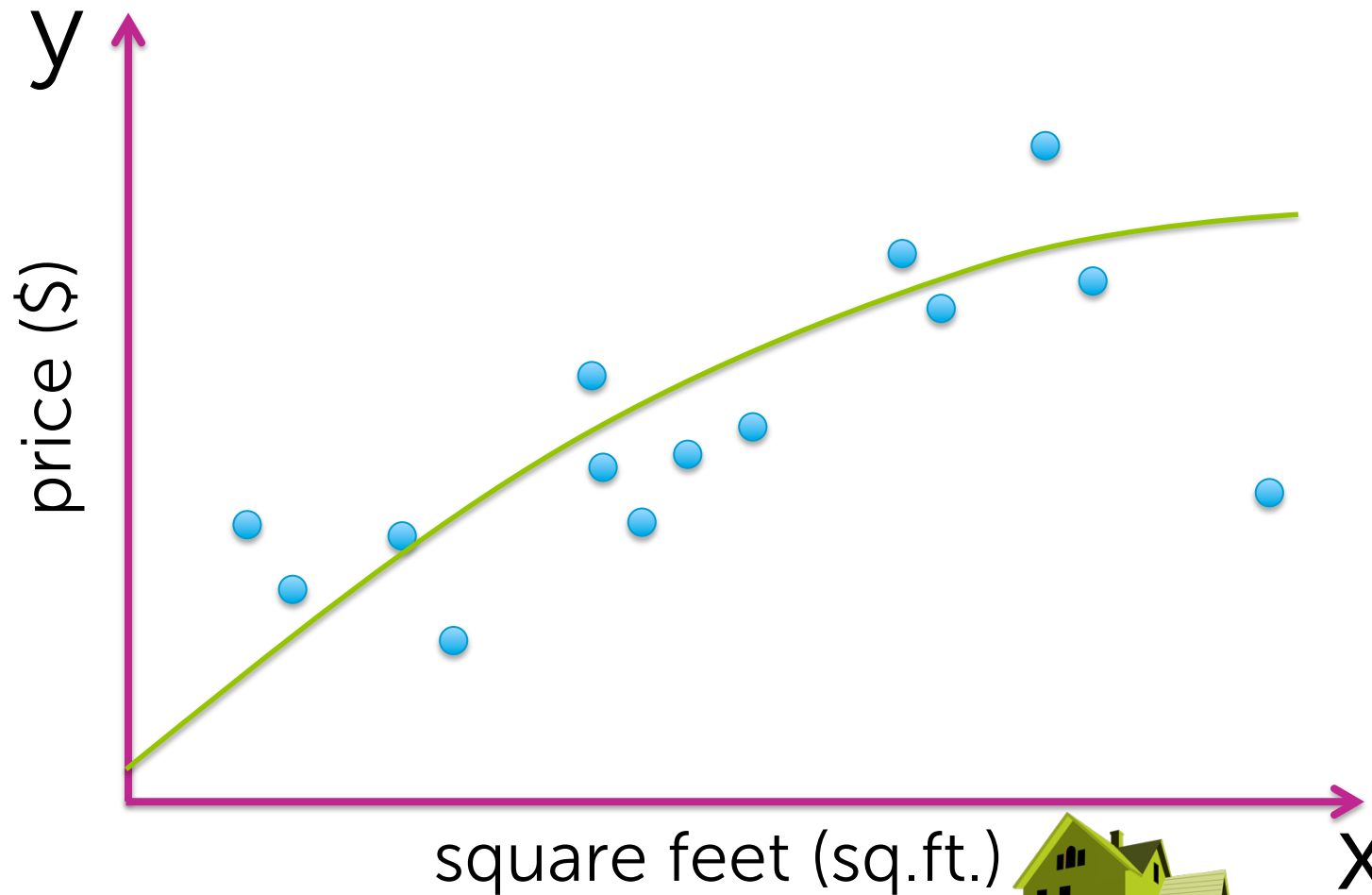


Fit data with a line or ... ?



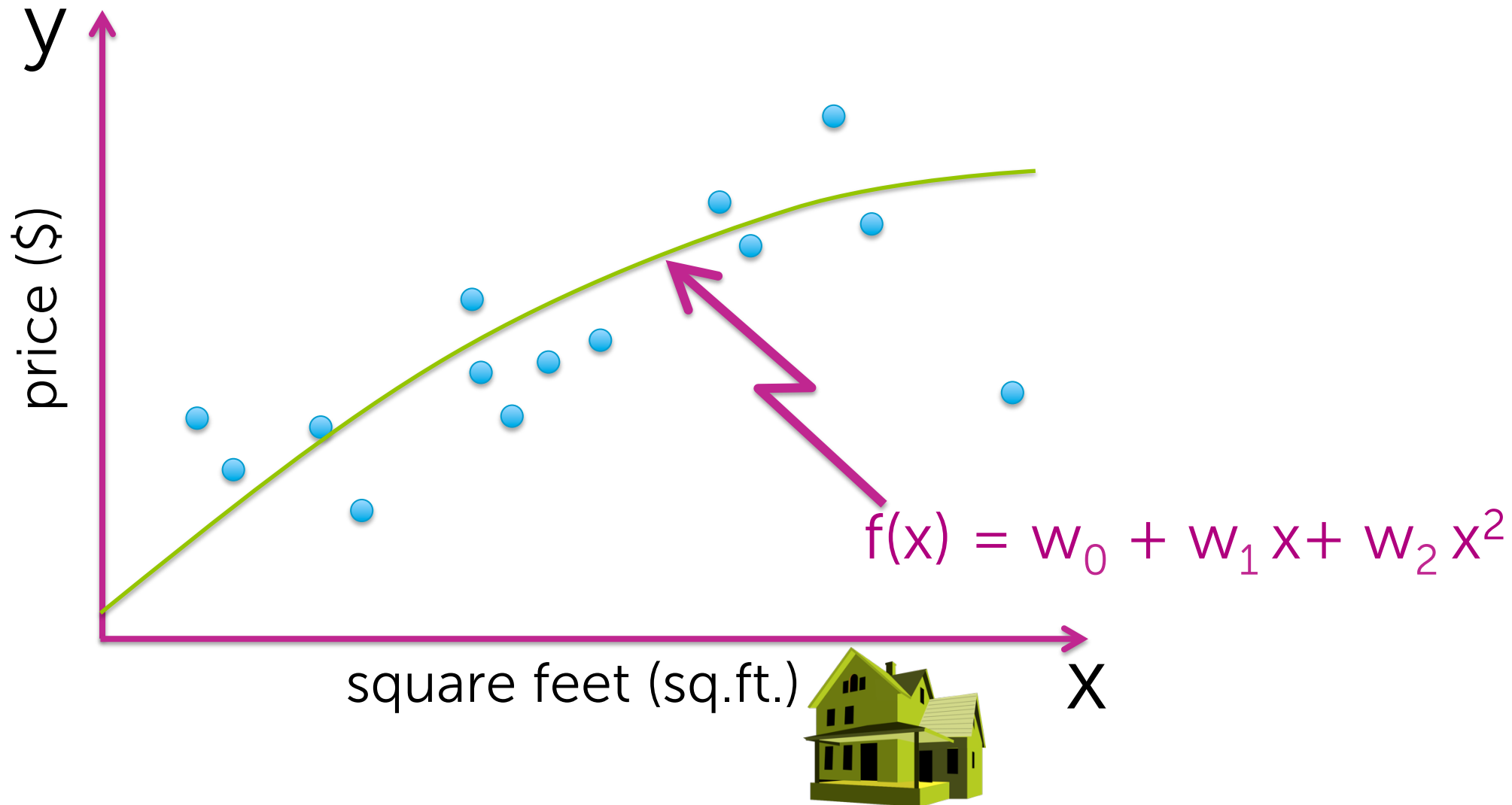
Dude, it's
not a linear
relationship!

What about a quadratic function?

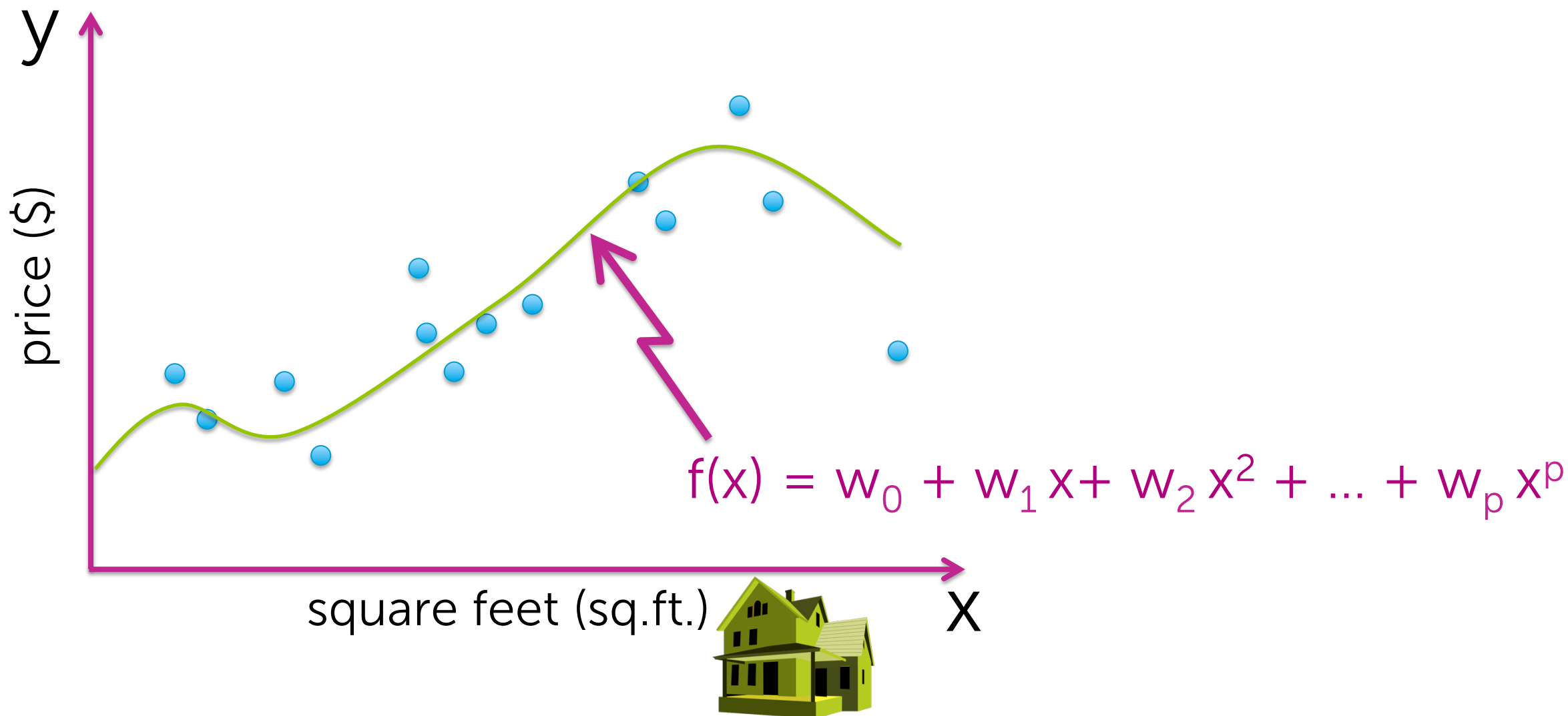


Dude, it's
not a linear
relationship!

What about a quadratic function?



Even higher order polynomial

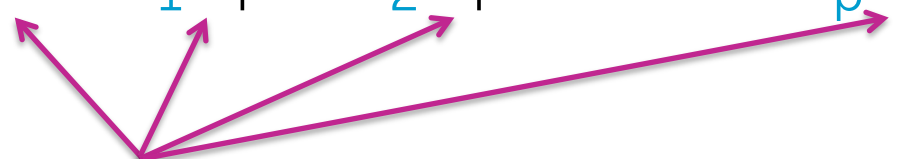


Polynomial regression

Model:

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

treat as different **features**



feature 1 = 1 (constant) parameter 1 = w_0

feature 2 = x parameter 2 = w_1

feature 3 = x^2 parameter 3 = w_2

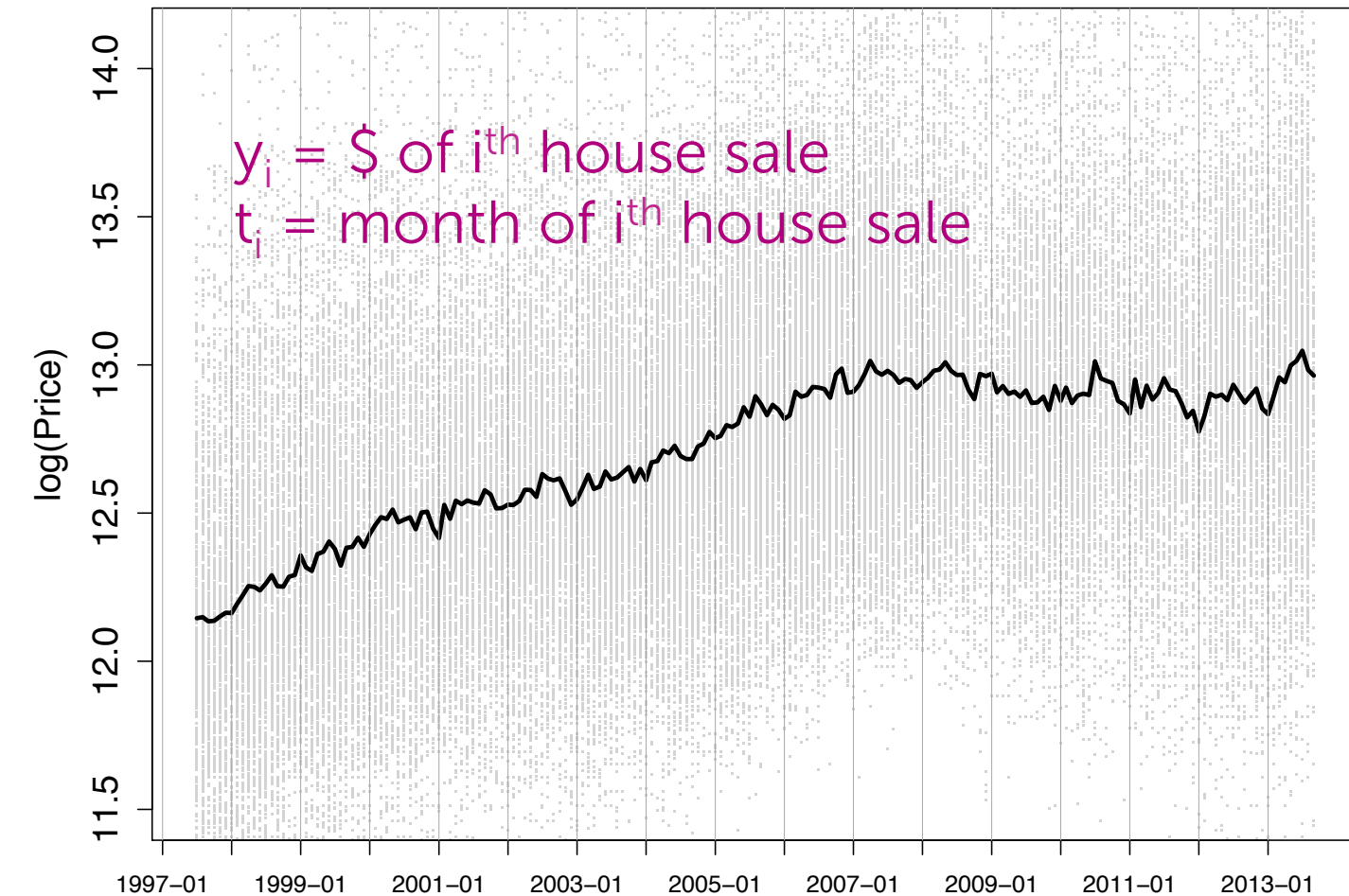
...

...

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

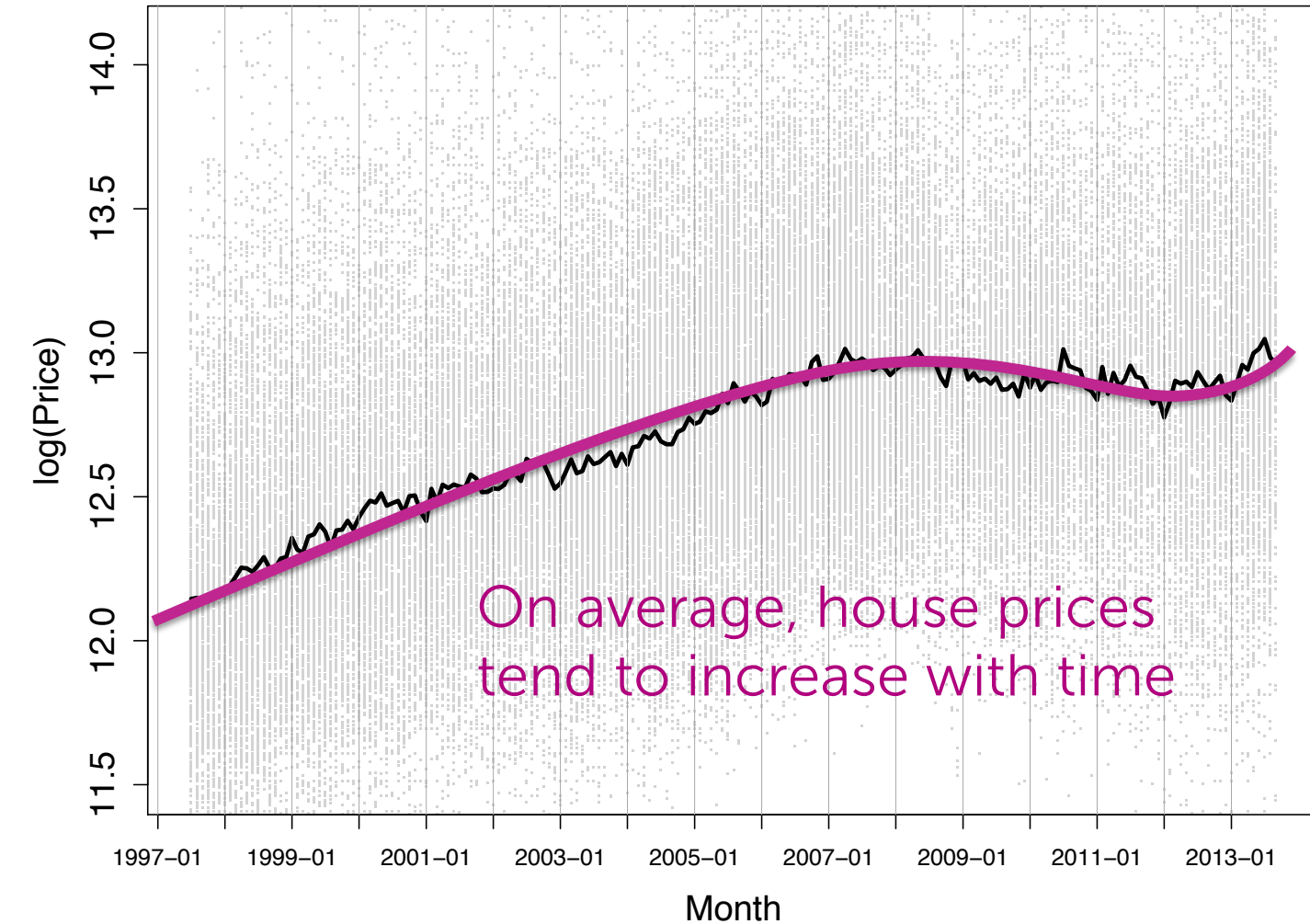
Other functions of one input

Motivating application: Detrending time series

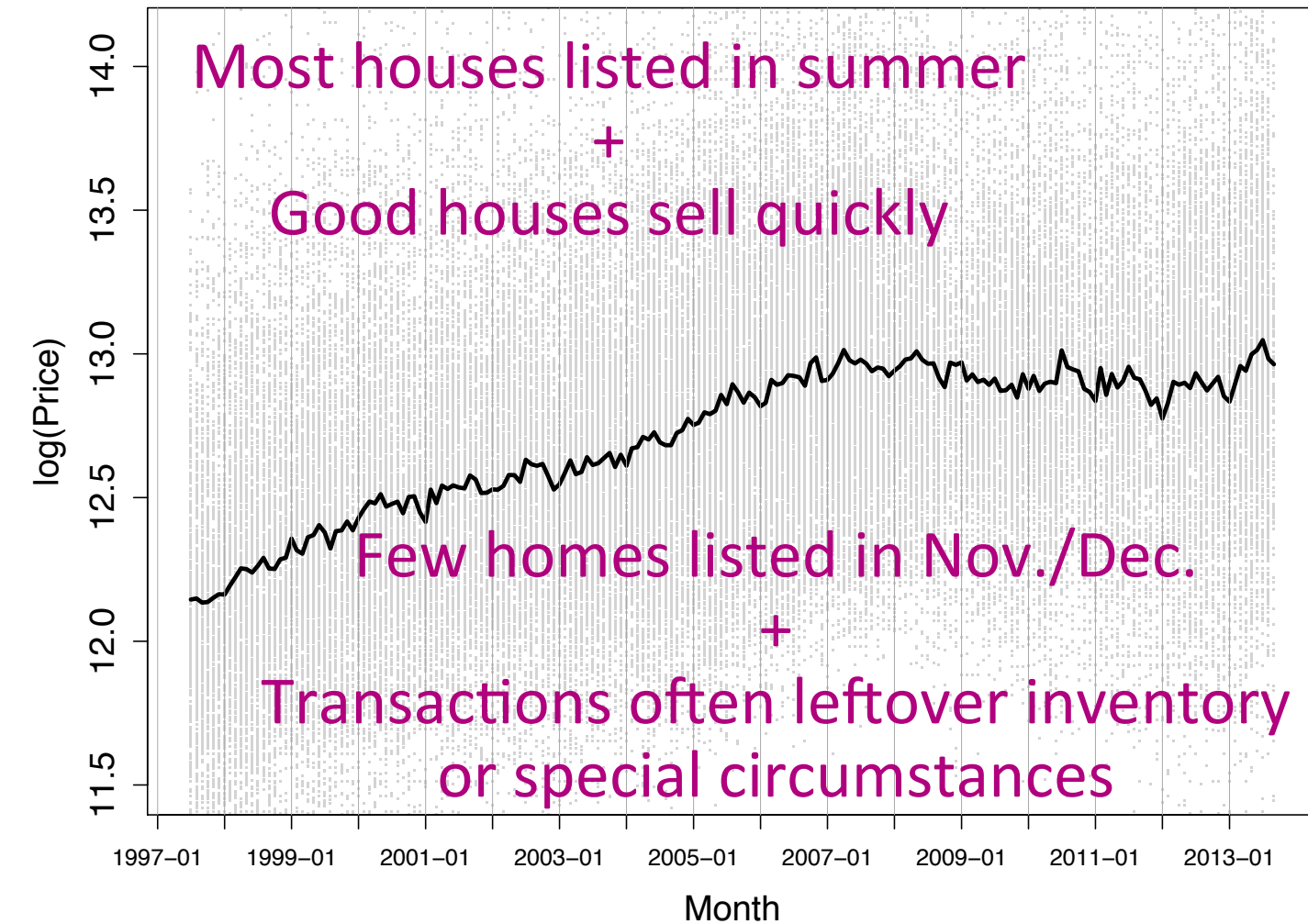


Month ← House sales recorded monthly

Trends over time



Seasonality



An example detrending

Model:

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

Unknown phase/shift

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,

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

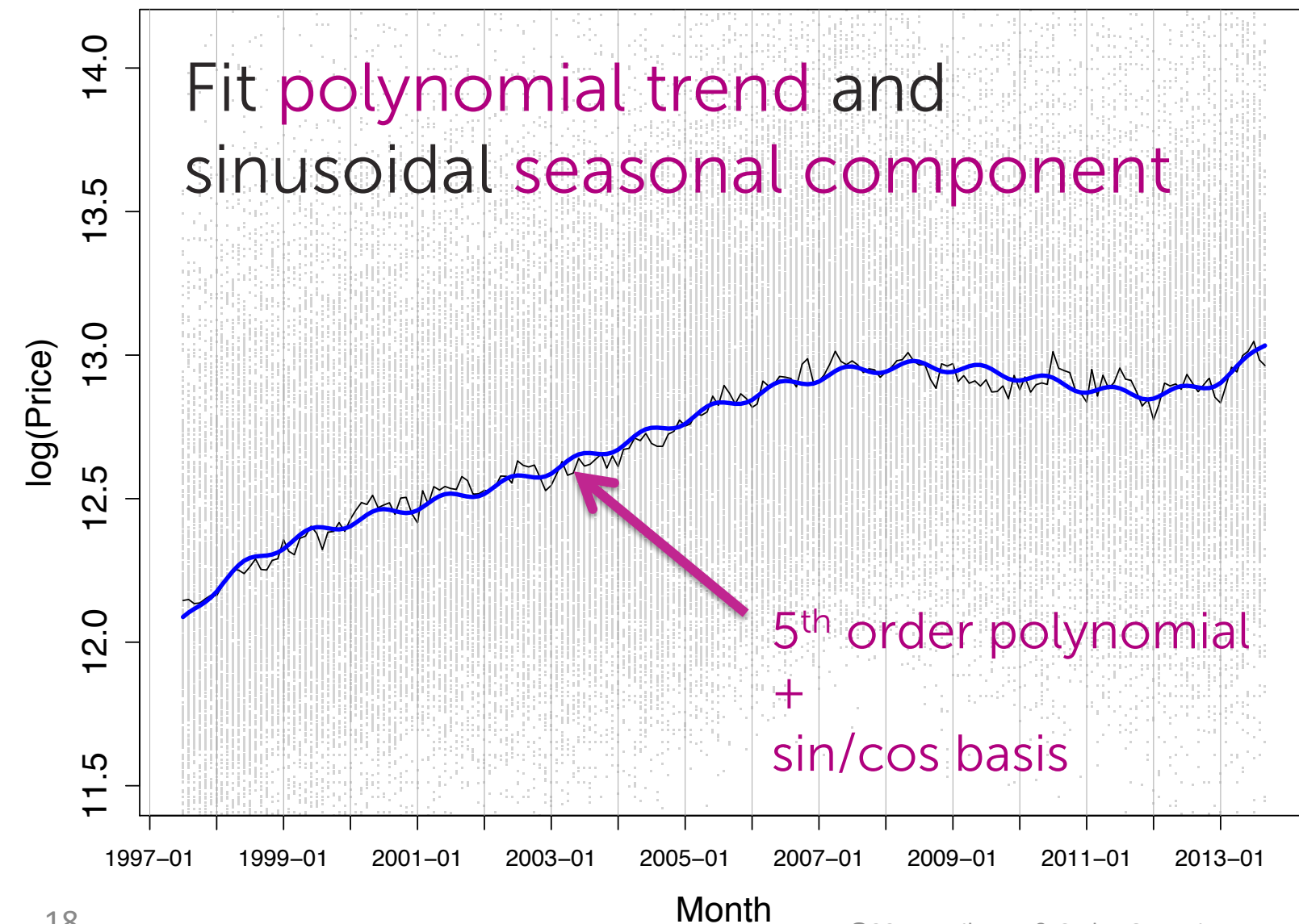
feature 1 = 1 (constant)

feature 2 = t

feature 3 = $\sin(2\pi t/12)$

feature 4 = $\cos(2\pi t/12)$

Detrended housing data

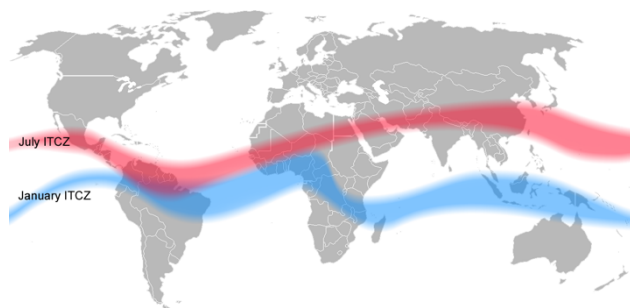


Zoom in...

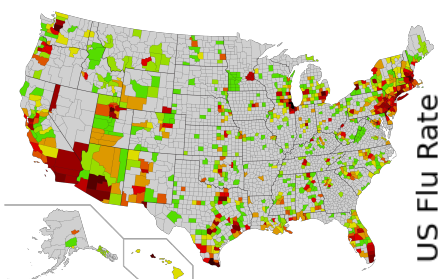
Fit polynomial trend and
sinusoidal seasonal component



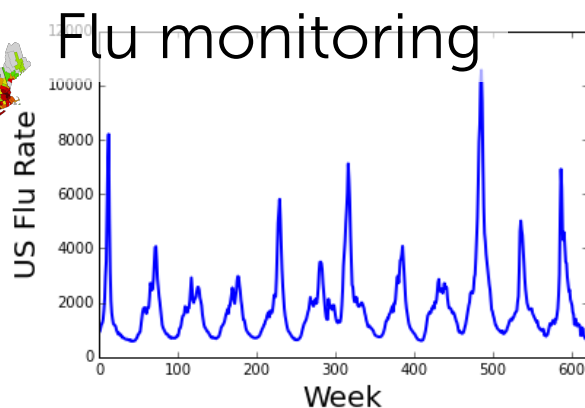
Other examples of seasonality



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



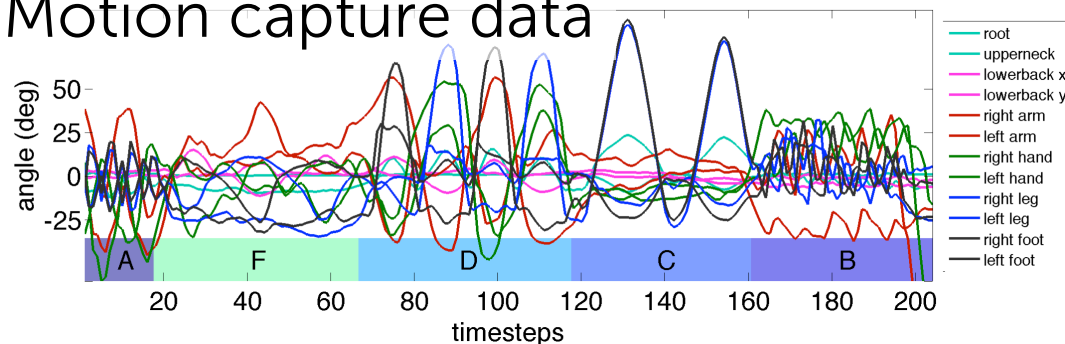
Flu monitoring



Demand forecasting
(e.g., jacket purchases)



Motion capture data



More generally...

Generic basis expansion

Model:

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

jth regression coefficient or weight

jth feature

Generic basis expansion

Model:

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

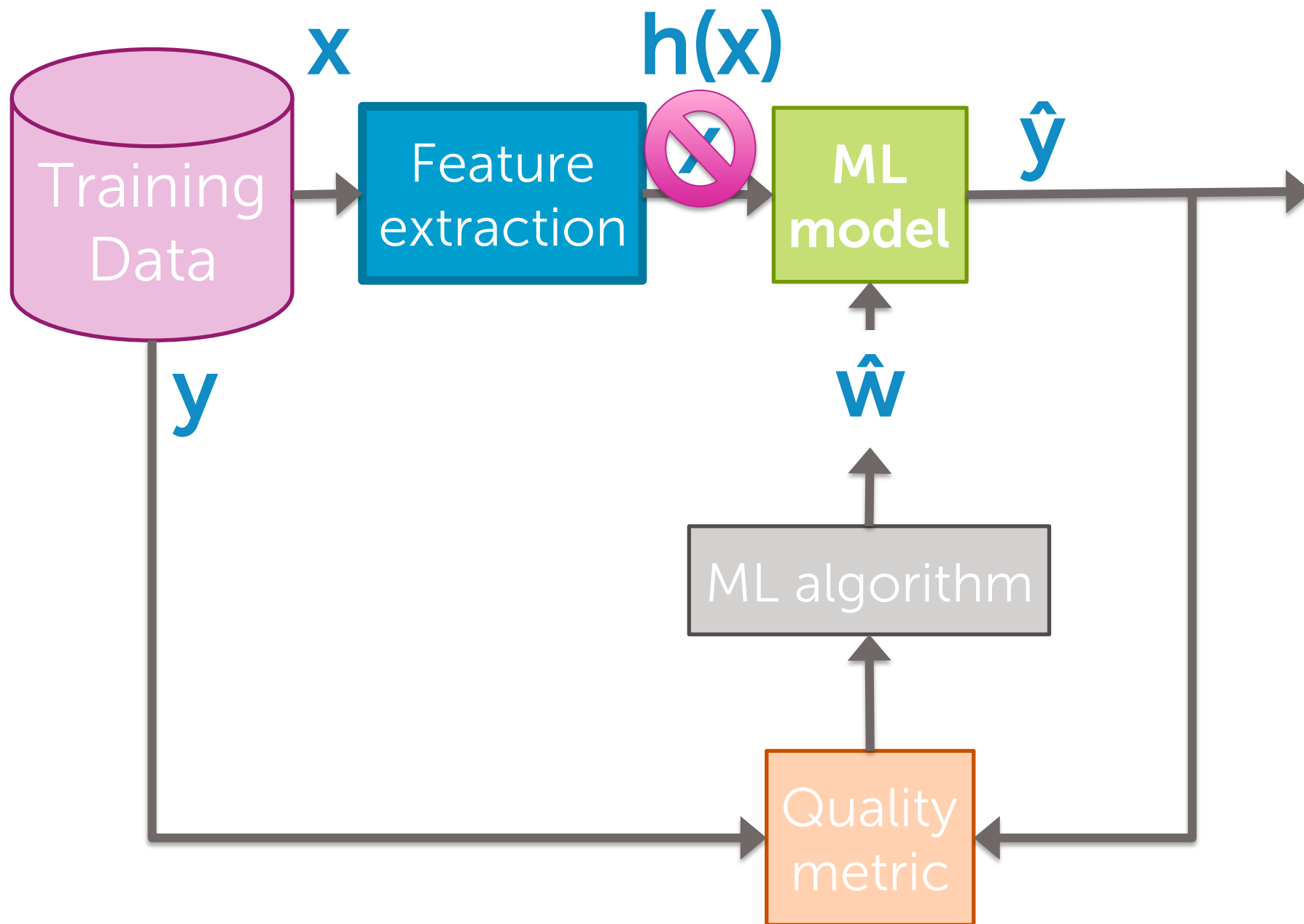
feature 1 = $h_0(x)$...often 1 (constant)

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

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

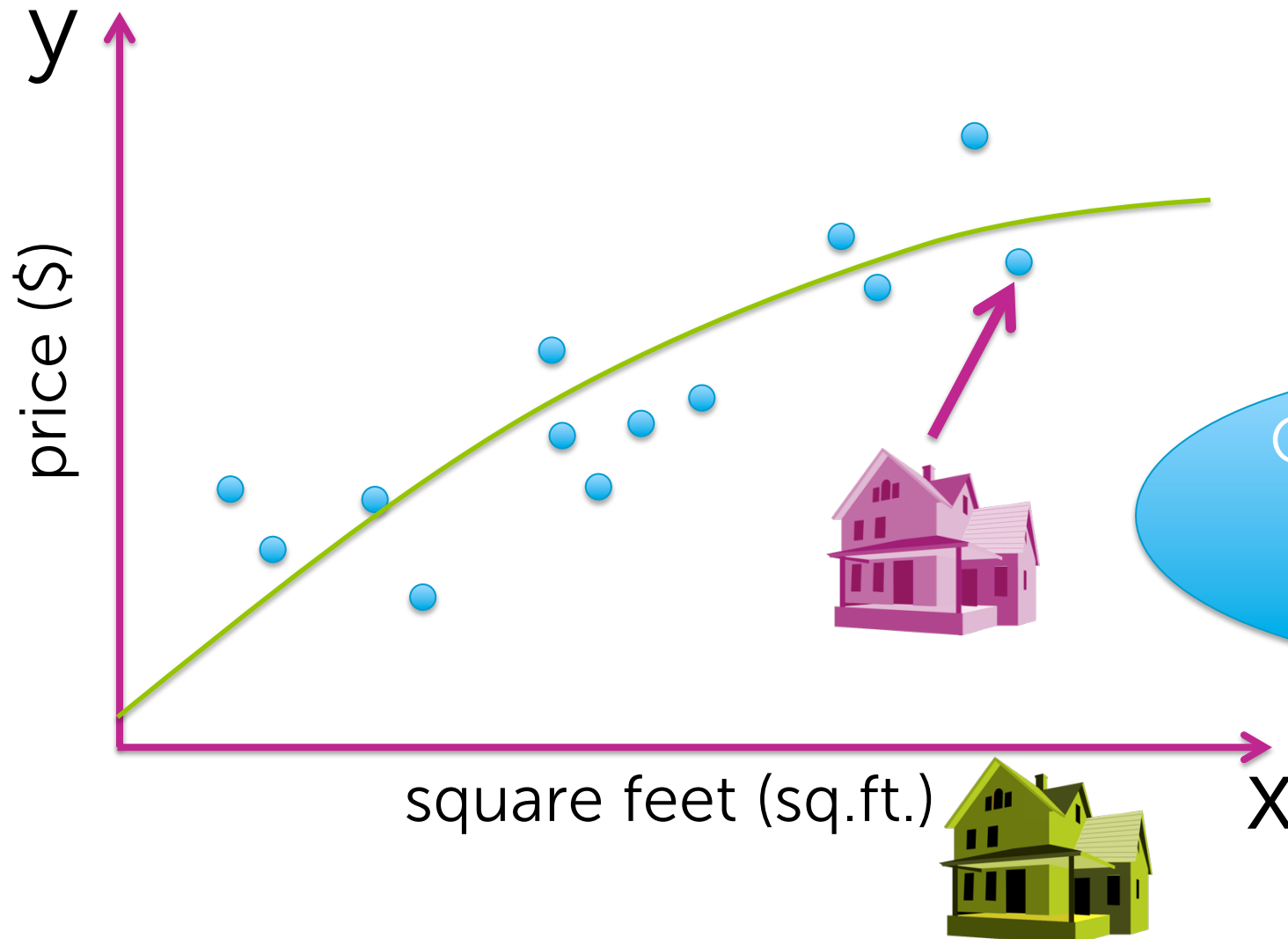
...

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



Incorporating multiple inputs

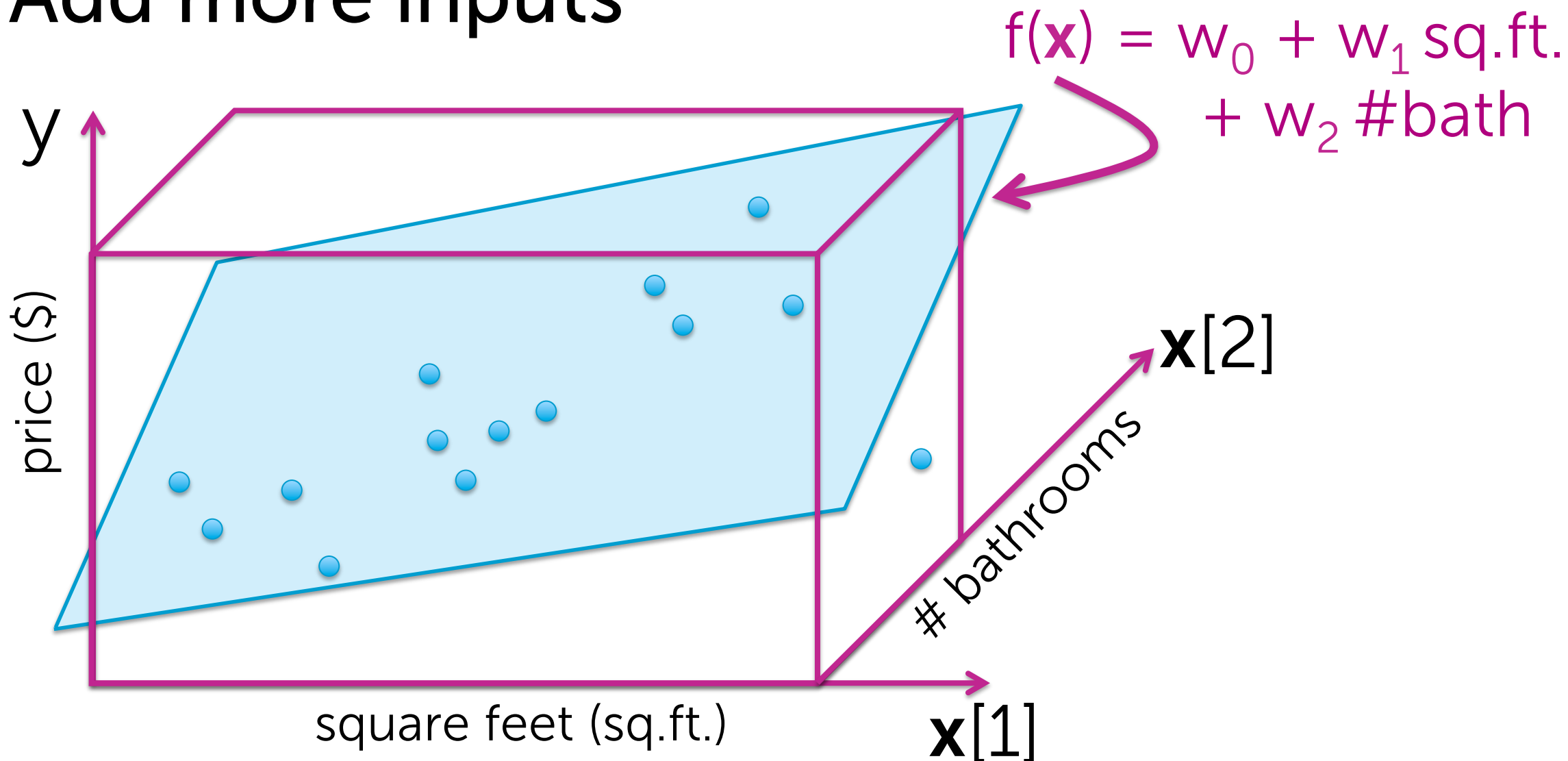
Predictions just based on house size



Only 1 bathroom!
Not same as my
3 bathrooms



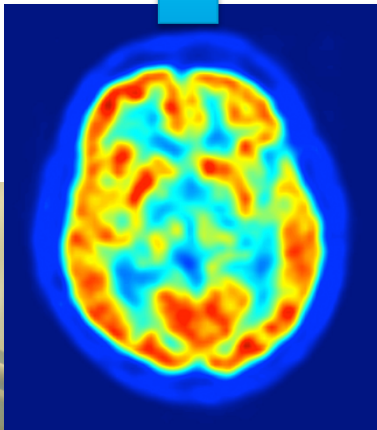
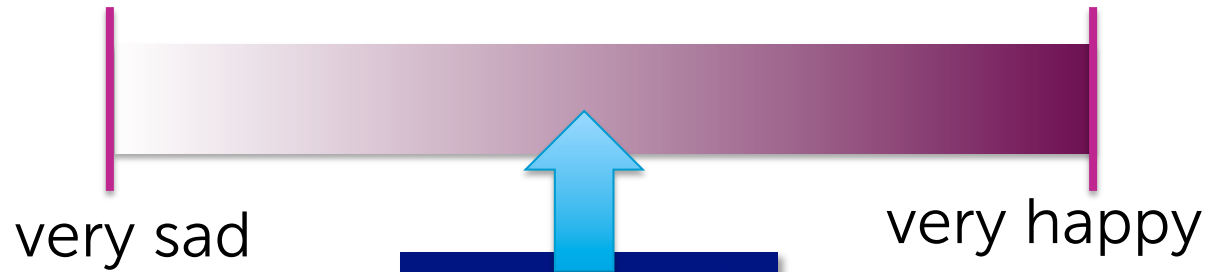
Add more inputs



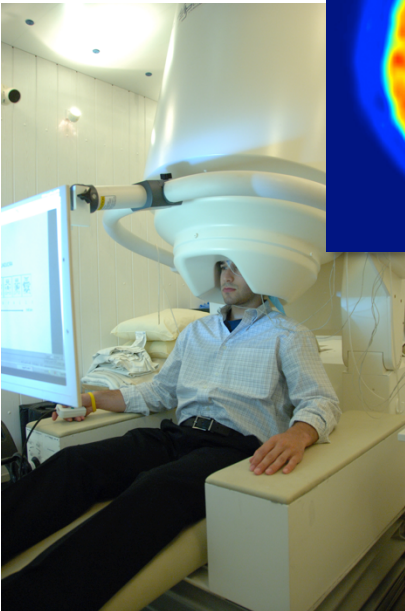
Many possible inputs

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

Reading your mind



Features are
brain region
intensities



General notation

Output: y  scalar

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

 d-dim vector

Notational conventions:

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

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

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

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

Simple hyperplane

Model:

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

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:

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

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, $\log(\mathbf{x}[7]) \mathbf{x}[2]$ = $\log(\text{\#bed}) \times \text{\#bath}$

...

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

More on notation

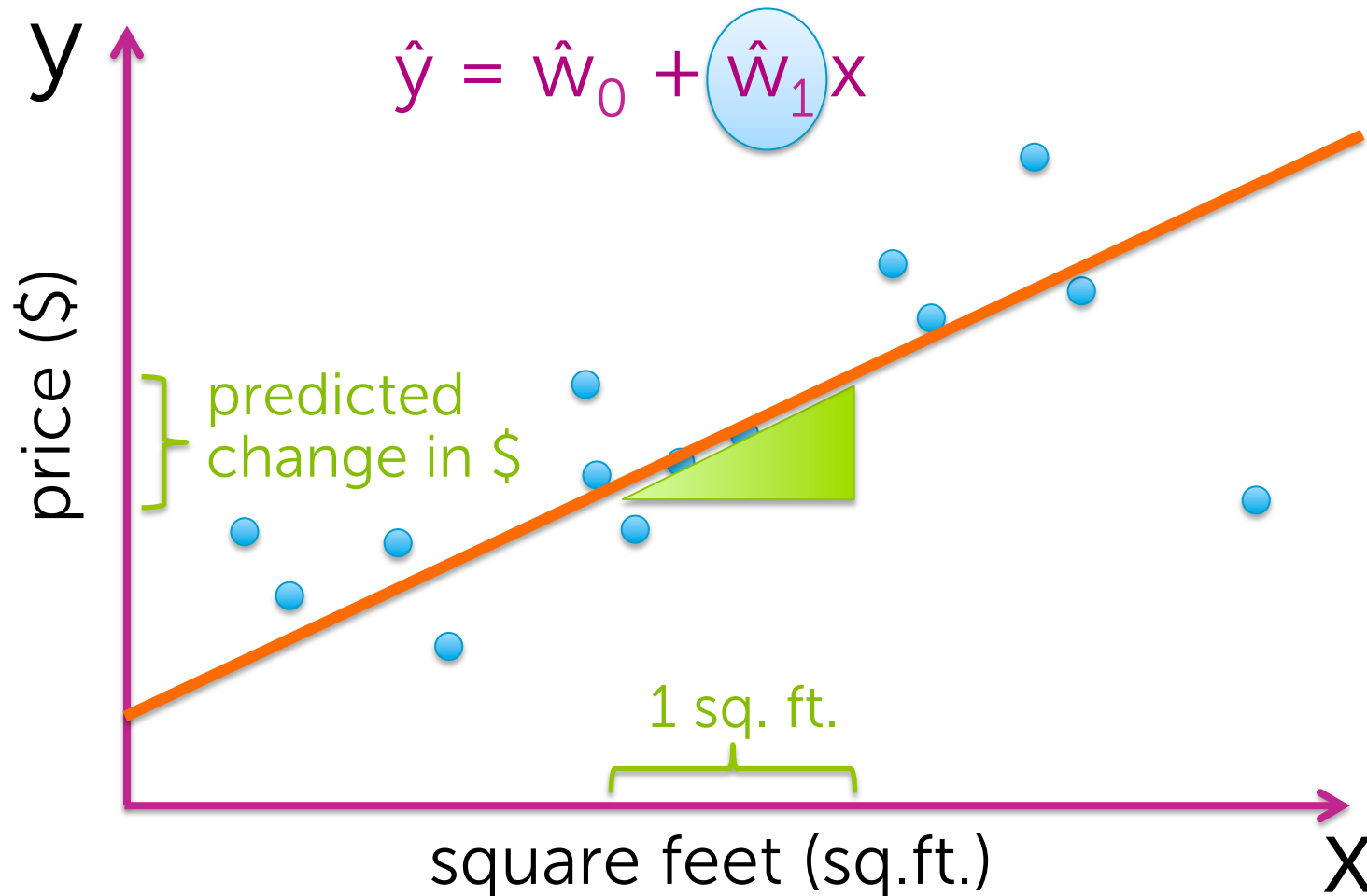
observations $(\mathbf{x}_i, y_i) : N$

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

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

Interpreting the fitted function

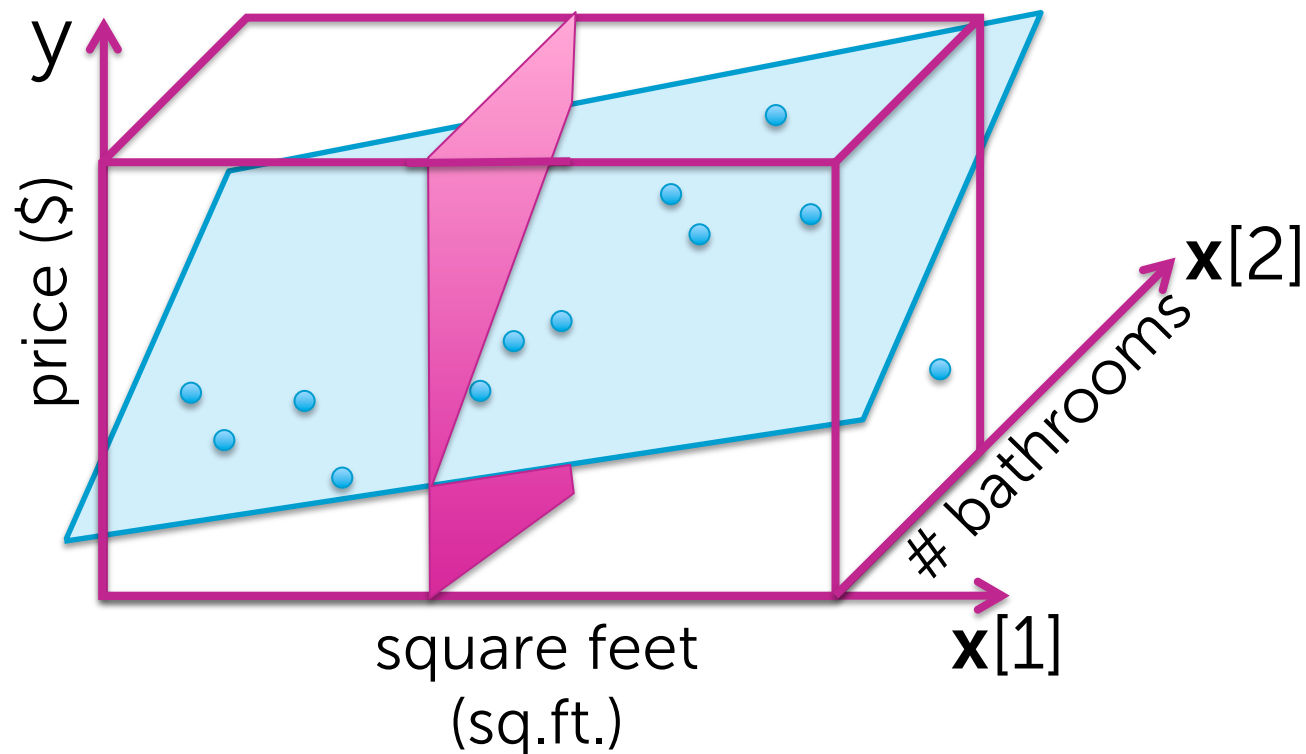
Interpreting the coefficients – Simple linear regression



Interpreting the coefficients – Two linear features

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

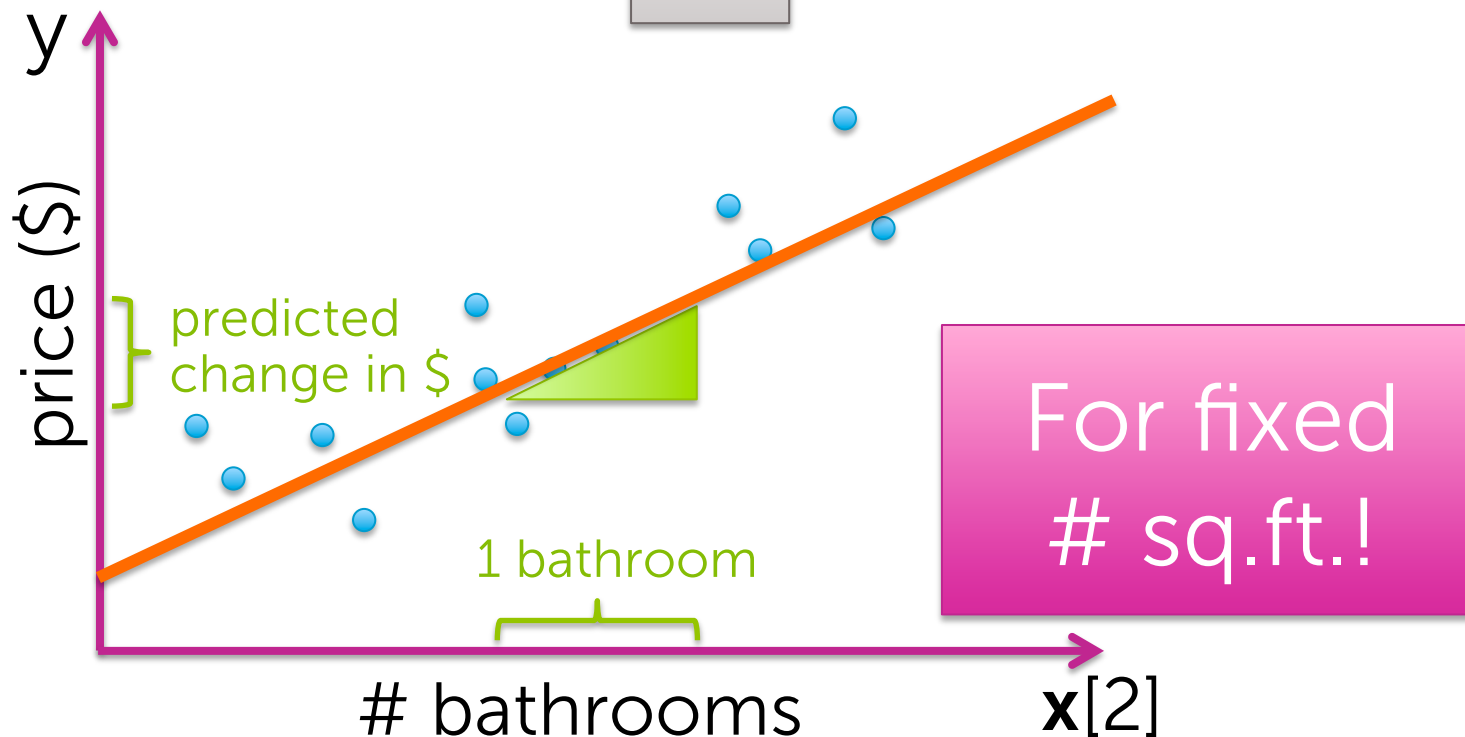
fix



Interpreting the coefficients – Two linear features

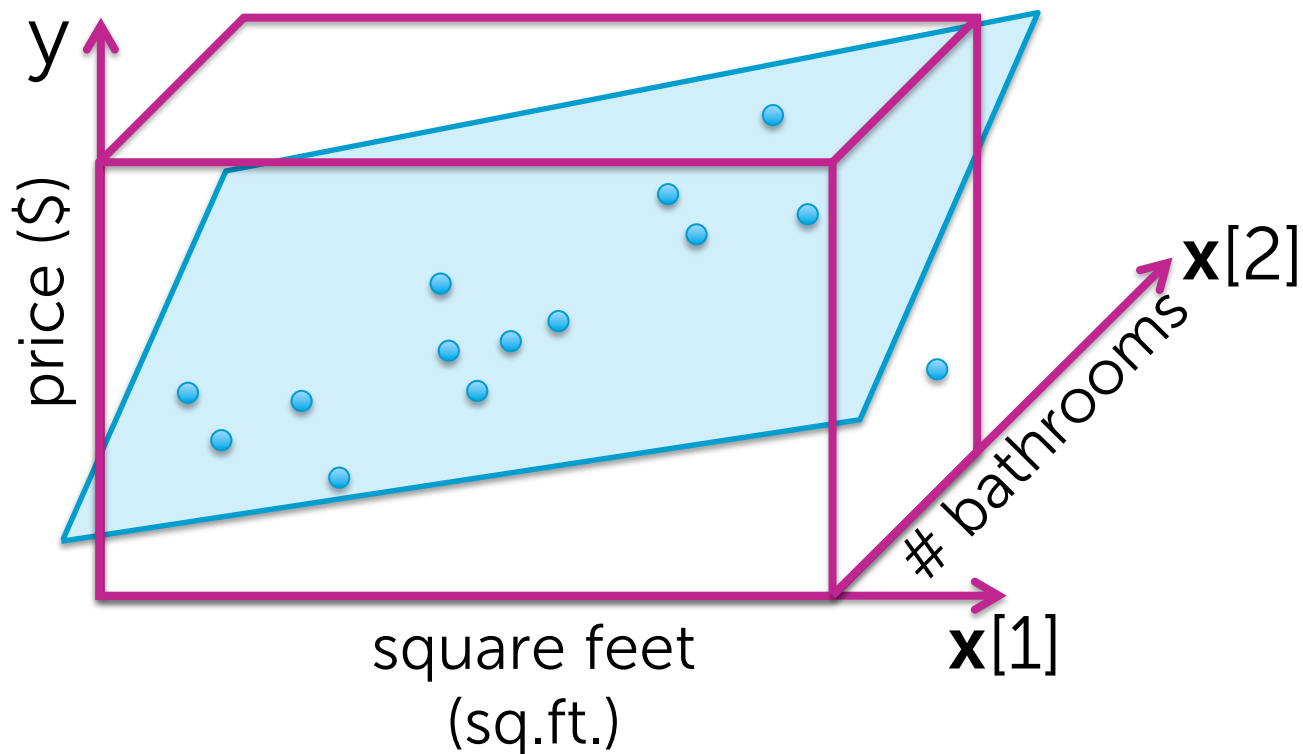
$$\hat{y} = \hat{w}_0 + \hat{w}_1 \mathbf{x}[1] + \hat{w}_2 \mathbf{x}[2]$$

fix



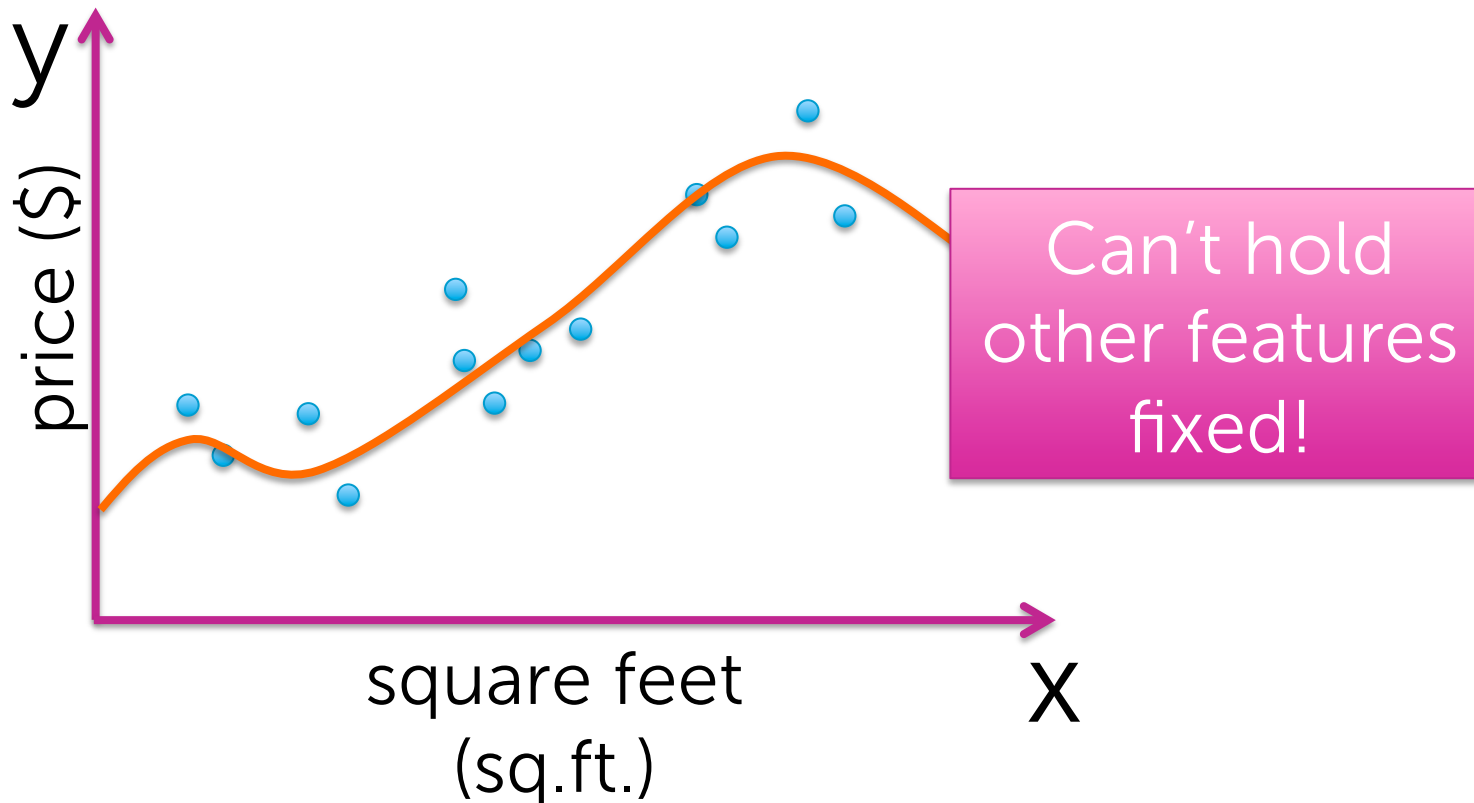
Interpreting the coefficients – Multiple linear features

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

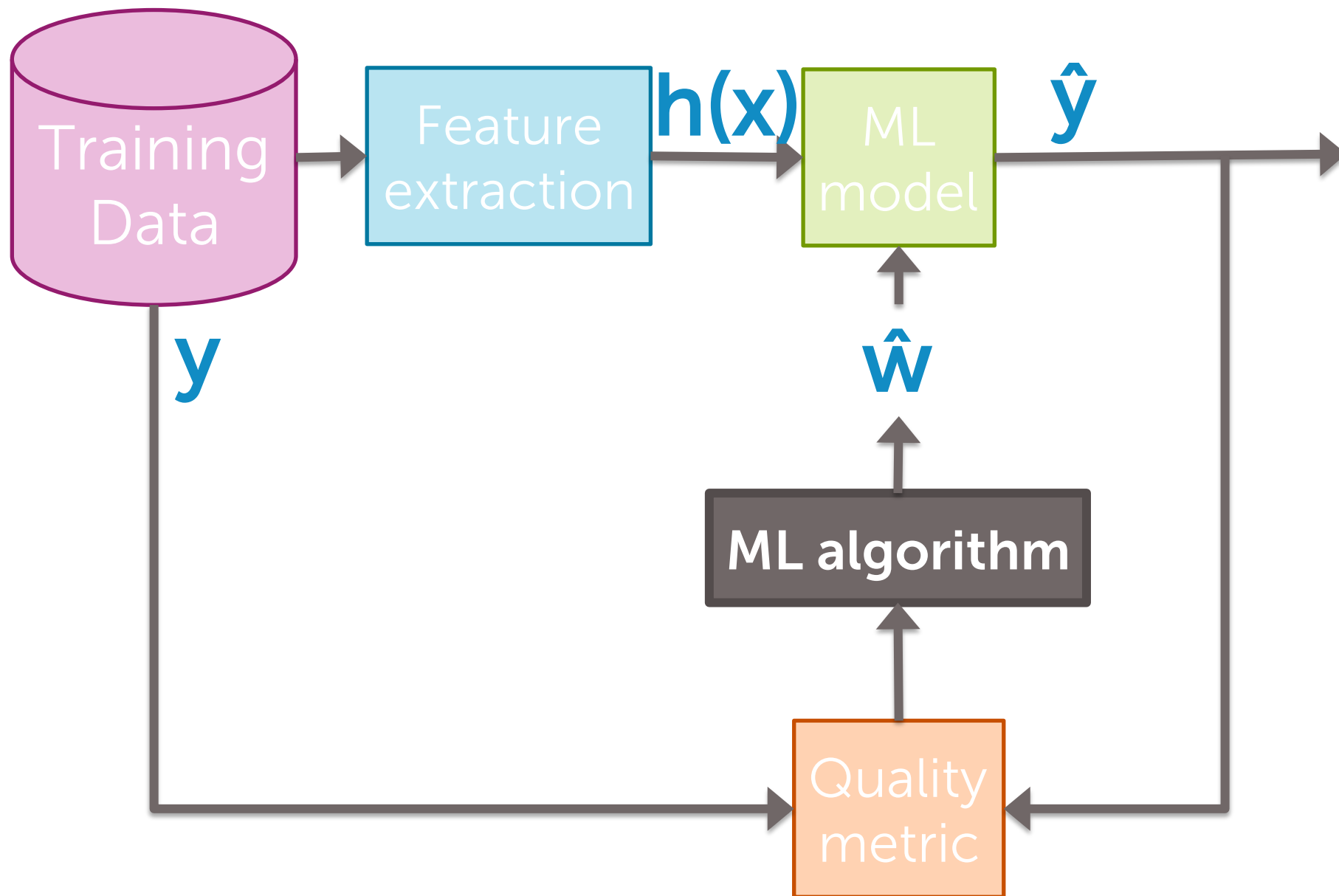


Interpreting the coefficients- Polynomial regression

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



Fitting D-dimensional curves



Step 1:

Rewrite the regression model

Rewrite in matrix notation

For observation i

$$y_i = \sum_{j=0}^D w_j h_j(\mathbf{x}_i) + \varepsilon_i$$

The diagram illustrates the conversion of the scalar equation for y_i into matrix notation. It shows three equivalent ways to represent the same calculation:

- Left side (Scalar expansion):** A pink box labeled y_i is equal to a row of blue boxes representing weights $w_0, w_1, w_2, \dots, w_D$. A blue bracket above this row is labeled w^T . Below the boxes, the expression $= w_0 h_0(x_i) + w_1 h_1(x_i) + \dots + w_D h_D(x_i) + \varepsilon_i$ is written. An arrow points to the first term $w_0 h_0(x_i)$ with the label "scalar". The final result is $= w^T h(x_i) + \varepsilon_i$.
- Middle side (Vector-matrix product):** A green vertical column of boxes represents the feature vector $h(x_i)$, with labels $h_0(x_i), h_1(x_i), h_2(x_i), \dots, h_D(x_i)$ to its right. A blue arrow points to the top of this column with the label $h(x_i)$. This is multiplied by a grey box representing the scalar ε_i .
- Right side (Matrix notation):** A green row of boxes represents the feature vector $h^T(x_i)$, with labels $h_0(x_i), h_1(x_i), \dots, h_D(x_i)$ below them. A blue bracket above this row is labeled $h^T(x_i)$. This is added to a green box labeled ε_i . This sum is then multiplied by a blue vertical column of boxes representing the weight vector w , with labels $w_0, w_1, \dots, w_D$ to its right. A blue bracket above this column is labeled w .

Rewrite in matrix notation

For all observations together

The diagram illustrates the matrix notation for a linear model across all observations. It shows three main components: the output vector Y , the hypothesis matrix H , the weight vector W , and the error vector E .

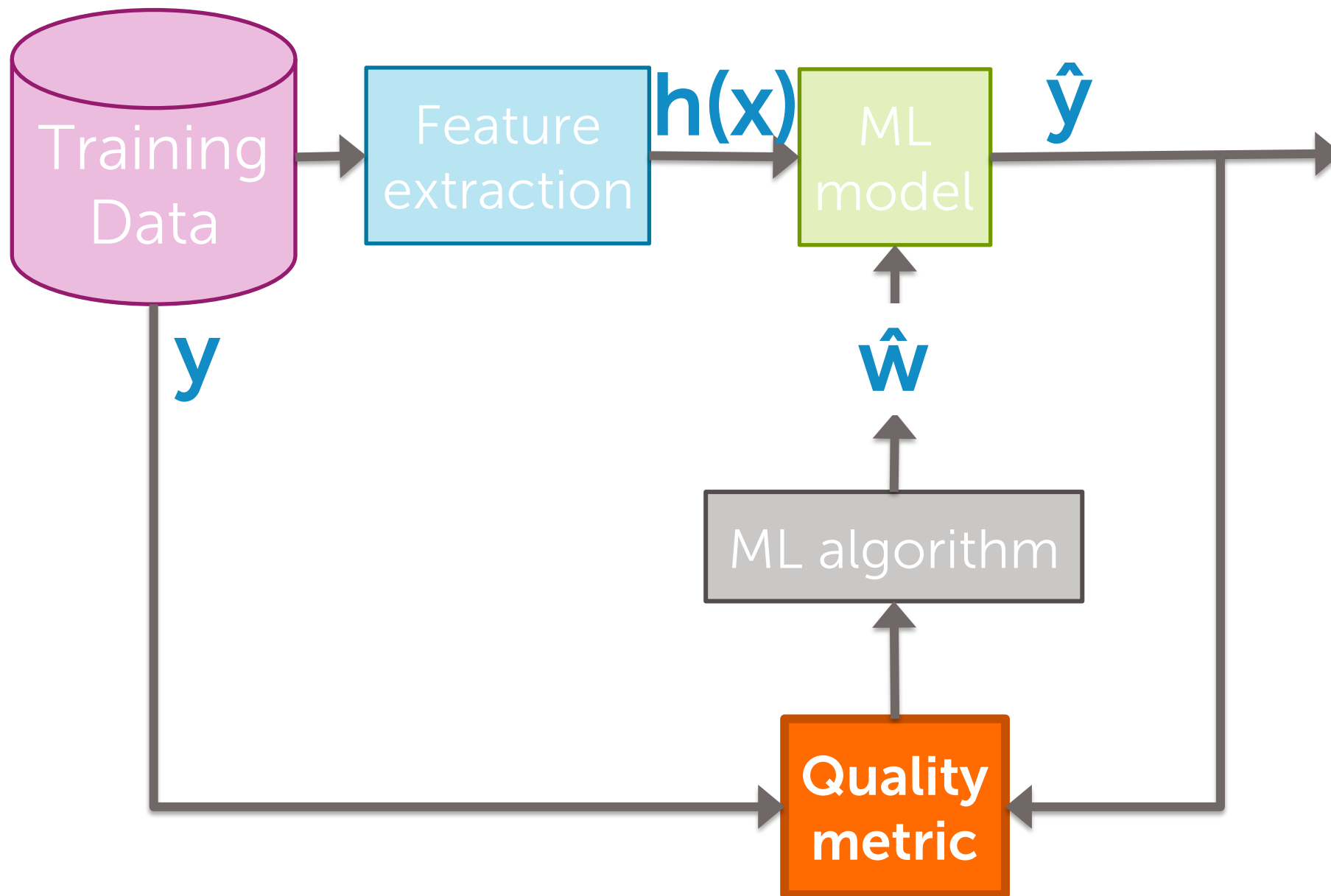
- Y (pink column vector): Contains the observed outputs $y_1, y_2, y_3, \dots, y_n$.
- H (green grid): The hypothesis matrix where each row represents an observation x_i and each column represents a feature $h_j(x)$. The first column is $h_0(x)$ and the last is $h_D(x)$.
- W (blue column vector): Contains the weights $w_0, w_1, w_2, \dots, w_D$.
- E (grey column vector): Contains the error terms $\epsilon_1, \epsilon_2, \epsilon_3, \dots, \epsilon_n$.

The equation is written as $Y = H W + E$.

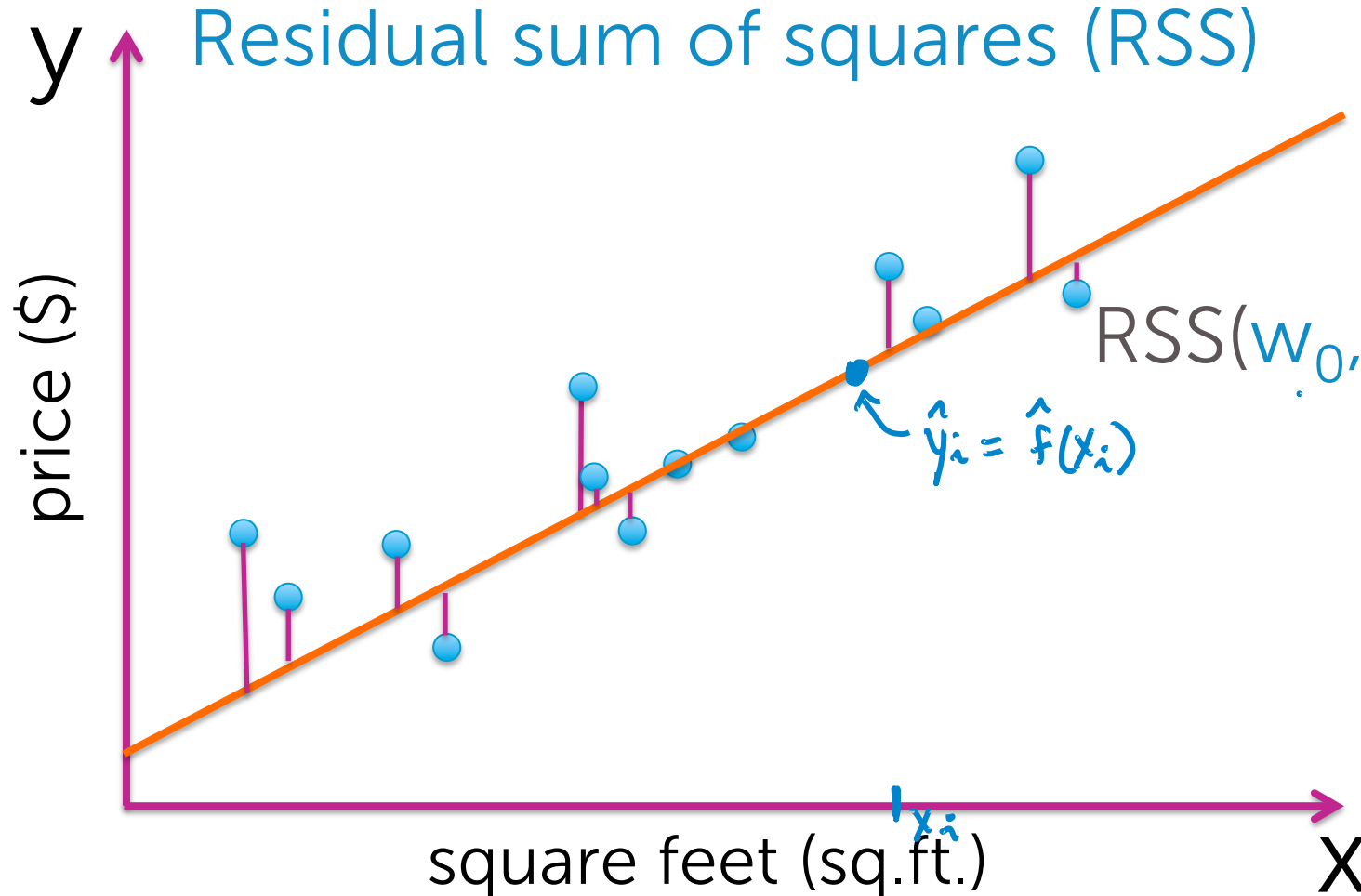
$$\Rightarrow \boxed{Y = H W + E}$$

Step 2:

Compute the cost

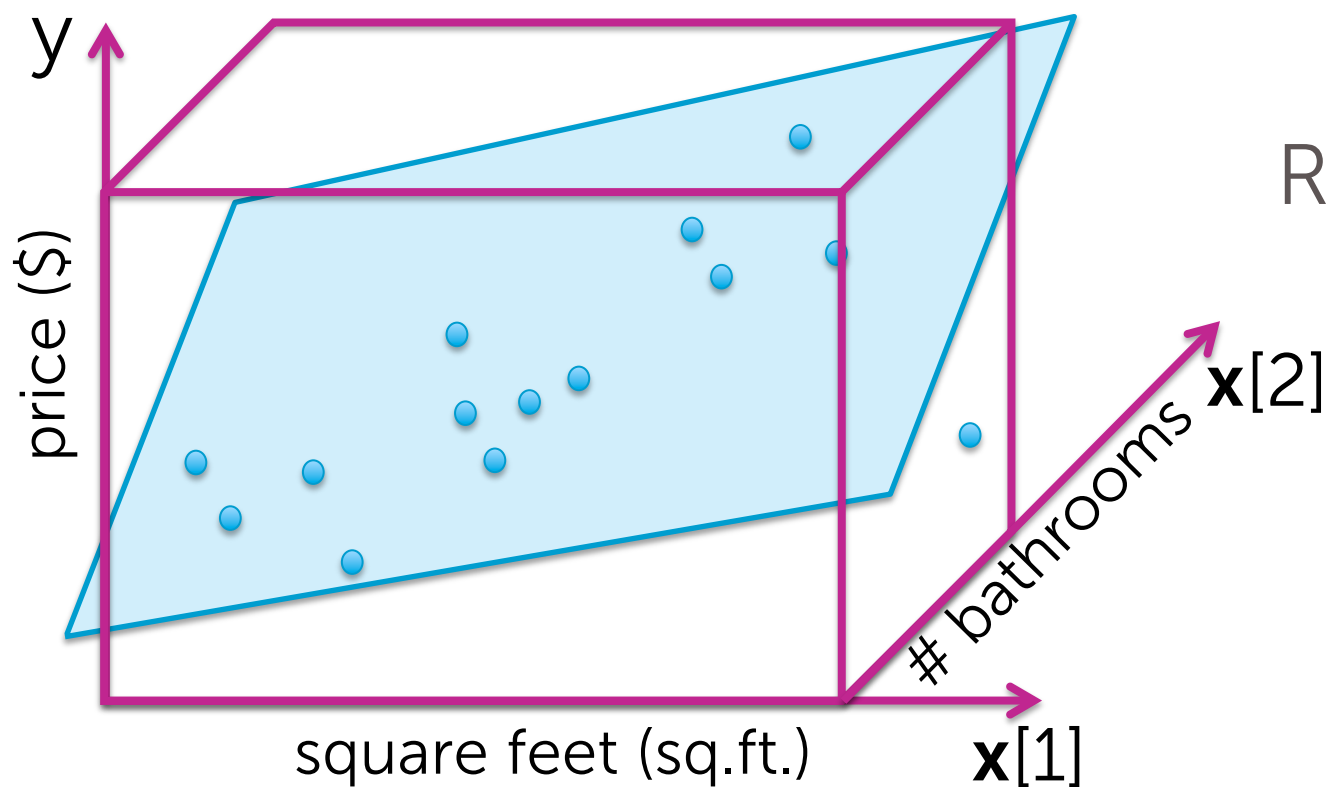


"Cost" of using a given line



$$\text{RSS}(w_0, w_1) = \sum_{i=1}^N (y_i - \underbrace{[w_0 + w_1 x_i]}_{\hat{y}_i(w_0, w_1)})^2$$

RSS for multiple regression



$$\text{RSS}(\underline{\mathbf{w}}) = \sum_{i=1}^N (y_i - \underbrace{h^T(x_i) \mathbf{w}}_{\hat{y}_i(\mathbf{w})})^2$$

Diagram illustrating the components of the equation:

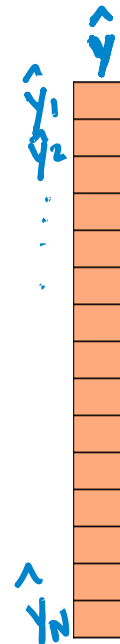
- $\hat{y}_i$ (orange box) is the predicted value for the i -th data point.
- $h^T(x_i)$ (green row vector) is the feature vector for the i -th data point, with components $h_0(x_i), h_1(x_i), \dots, h_D(x_i)$.
- $\mathbf{w}$ (blue column vector) is the weight vector, with components $w_0, w_1, w_2, \dots, w_D$.

RSS in matrix notation

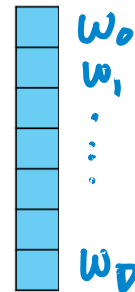
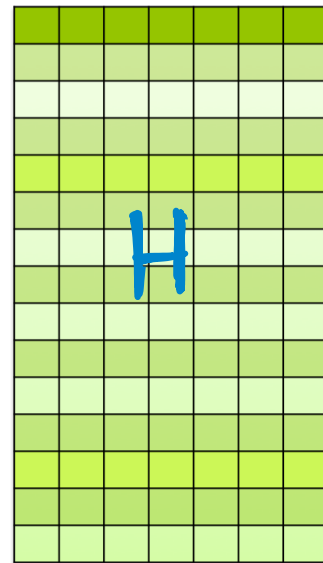
$$\text{RSS}(\mathbf{w}) = \sum_{i=1}^N (y_i - h(\mathbf{x}_i)^T \mathbf{w})^2$$

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

Why? (part 1)



=

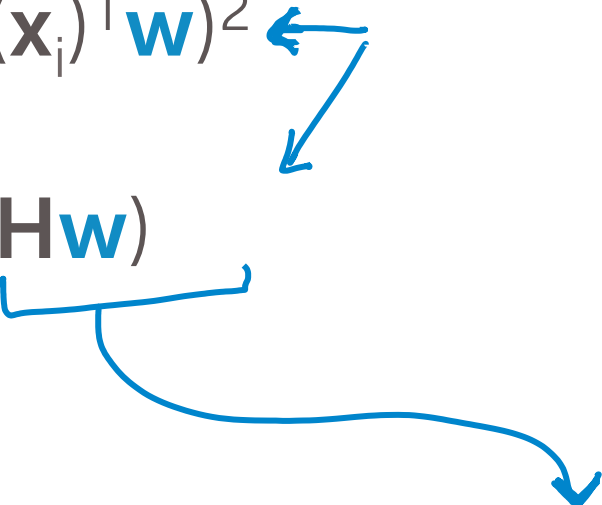


$$\hat{\mathbf{y}} = \mathbf{H}\mathbf{w}$$

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

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

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 2)

residual ₁	residual ₂	residual ₃	...	residual _N
residual ₁	residual ₂	residual ₃	...	residual _N

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

Step 3:

Take the gradient

Gradient of RSS

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

Why? By analogy to 1D case:

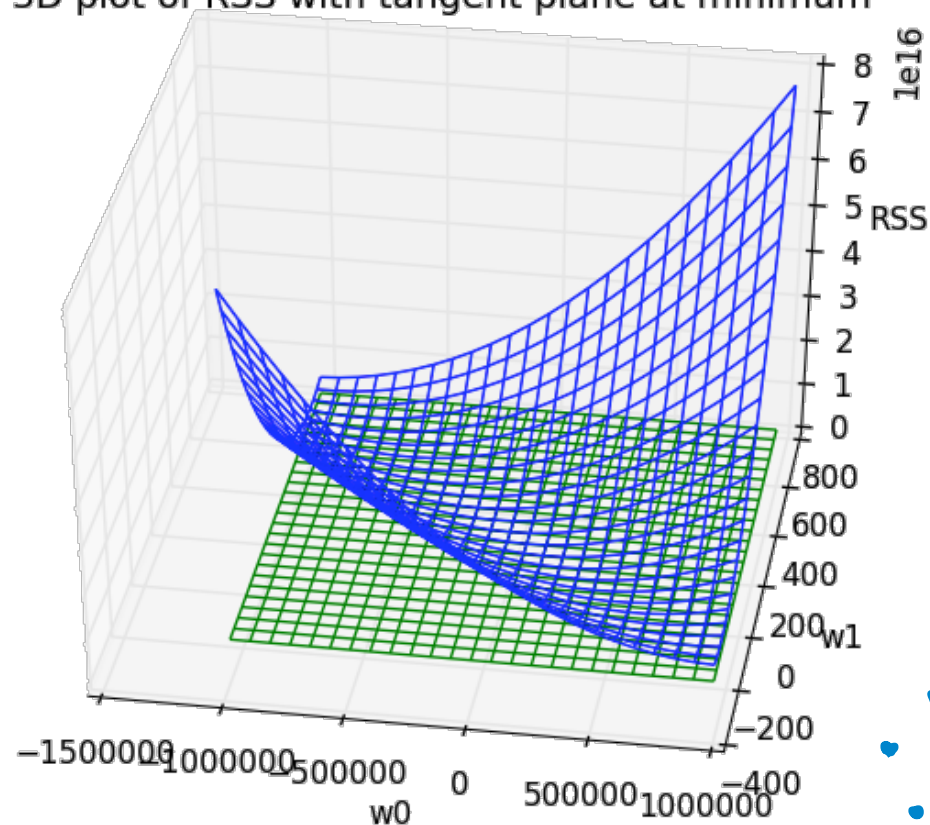
$$\begin{aligned}\frac{d}{dw} (y-hw)(y-hw) &= \frac{d}{dw} (y-hw)^2 = 2 \cdot (y-hw)' (-h) \\ &= -2h(y-hw)\end{aligned}$$

scalars

Step 4, Approach 1:
Set the gradient = 0

Closed-form solution

3D plot of RSS with tangent plane at minimum



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

Solve for $\mathbf{w}$:

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

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

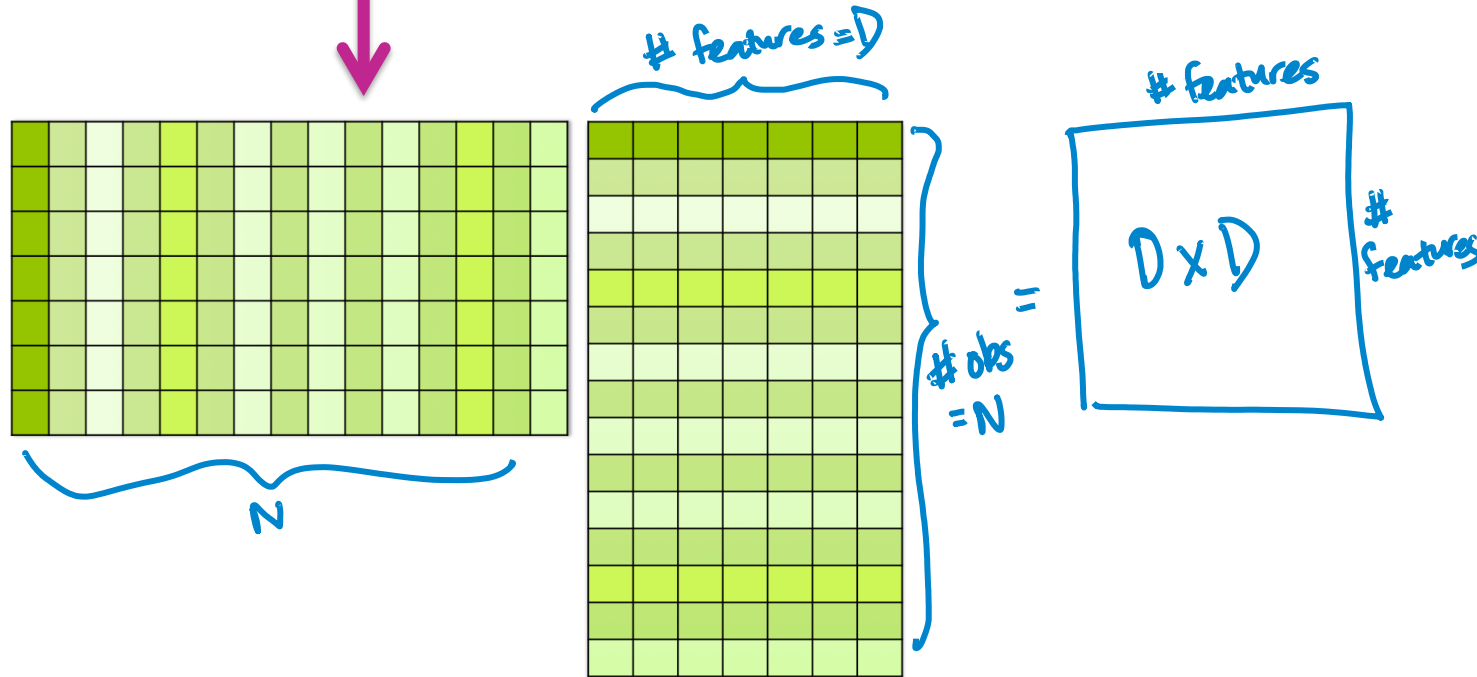
$$\underbrace{(\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T\mathbf{H}}_{\mathbf{I}}\hat{\mathbf{w}} = (\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T\mathbf{y}$$

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

- $\mathbf{A}^{-1}\mathbf{A} = \mathbf{I}$
- $\mathbf{I}\mathbf{v} = \mathbf{v}$
- $\mathbf{I}\mathbf{v} = \mathbf{v}$

Closed-form solution

$$\hat{\mathbf{W}} = (\underbrace{\mathbf{H}^T \mathbf{H}}_{\substack{\text{\# features} \\ = D}})^{-1} \mathbf{H}^T \mathbf{y}$$



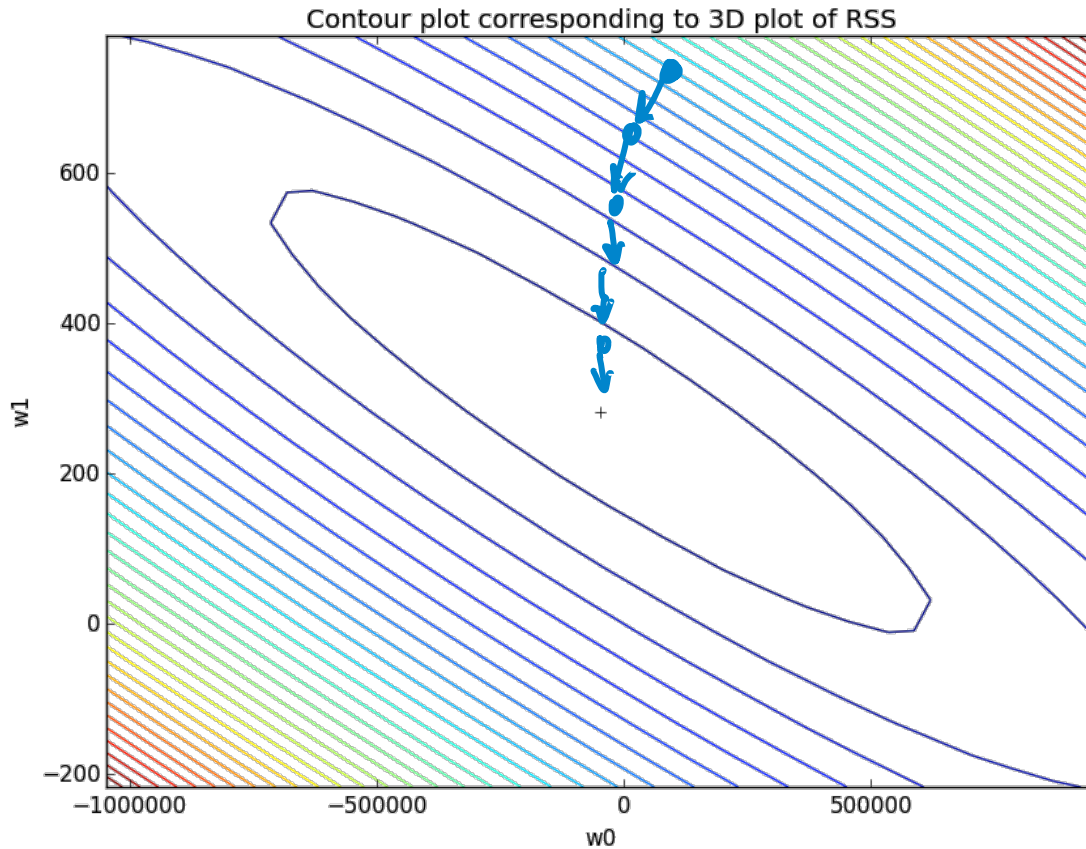
Invertible if:
In most cases is $N > D$
really, # of linearly ind. observations

Complexity of inverse:

$$O(D^3)$$

Step 4, Approach 2: Gradient descent

Gradient descent



while not converged

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

$$\leftarrow \mathbf{w}^{(t)} + 2\eta \mathbf{H}^T(\mathbf{y} - \underbrace{\mathbf{H}\mathbf{w}^{(t)}}_{\hat{\mathbf{y}}(\mathbf{w}^{(t)})})$$

Feature-by-feature update

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

$$= \sum_{i=1}^N (y_i - w_0 h_0(x_i) - w_1 h_1(x_i) - \dots - w_D h_D(x_i))^2$$

Partial with respect to w_j

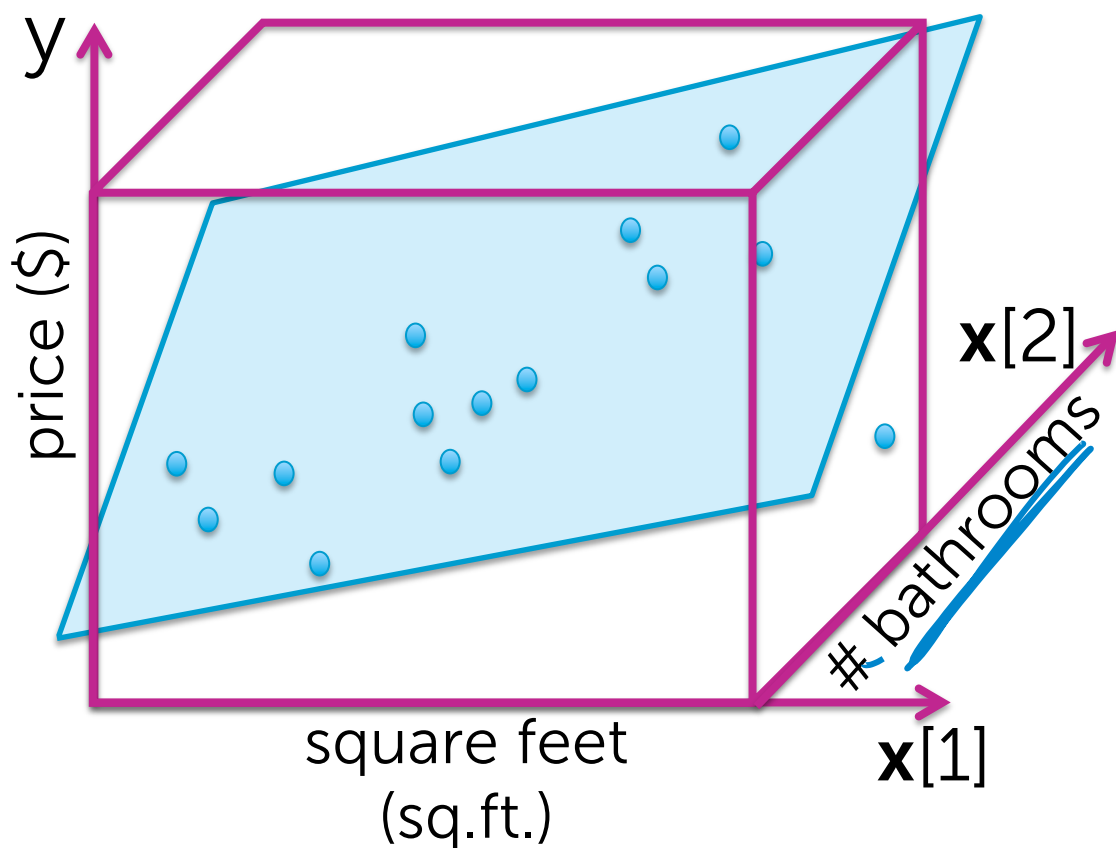
$$\sum_{i=1}^N 2 (y_i - w_0 h_0(x_i) - w_1 h_1(x_i) - \dots - w_D h_D(x_i)) \cdot (-h_j(x_i))$$

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

Update to j^{th} feature weight:

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

Interpreting elementwise

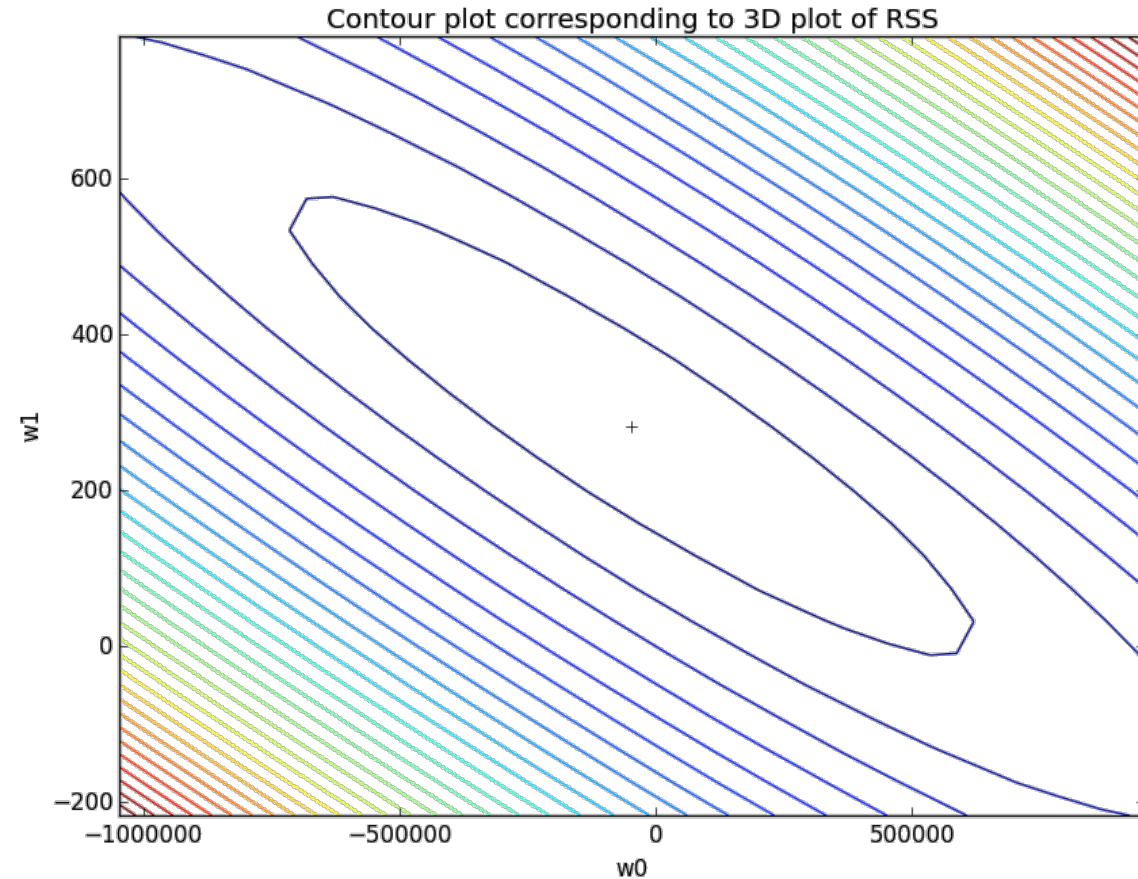


Update to j^{th} feature weight:

$$w_j^{(t+1)} \leftarrow w_j^{(t)} + 2\eta \sum_{i=1}^N h_i(\mathbf{x}_i) (y_i - \hat{y}_i(\mathbf{w}^{(t)}))$$

If underestimating impact of # bath ($\hat{w}_j^{(t)}$ is too small)
 then $(y_i - \hat{y}_i(\mathbf{w}^{(t)}))$ 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 $j = 0, \dots, D$
 $\text{partial}[j] = -2 \sum_{i=1}^N h_j(\mathbf{x}_i)(y_i - \hat{y}_i(\mathbf{w}^{(t)}))$
 $\mathbf{w}_j^{(t+1)} \leftarrow \mathbf{w}_j^{(t)} - \eta \text{partial}[j]$
 $t \leftarrow t + 1$

Note: The norm in the while loop is defined as $\sqrt{\text{partial}[0]^2 + \dots + \text{partial}[D]^2}$.

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