- Join gradescope
  - there are instructions on piazza
- lots of info on website

Linear Regression: Model and Algorithms

CSE 446

Slides by Emily Fox (with minor modifications)
Presented by Anna Karlin
April 3, 2019

Linear regression: The model

How much is this house worth?

How much is this house worth?

Data

input *output*

$(x_1 = \text{sq.ft.}, y_1 = \$)$

$(x_2 = \text{sq.ft.}, y_2 = \$)$

$(x_3 = \text{sq.ft.}, y_3 = \$)$

$(x_4 = \text{sq.ft.}, y_4 = \$)$

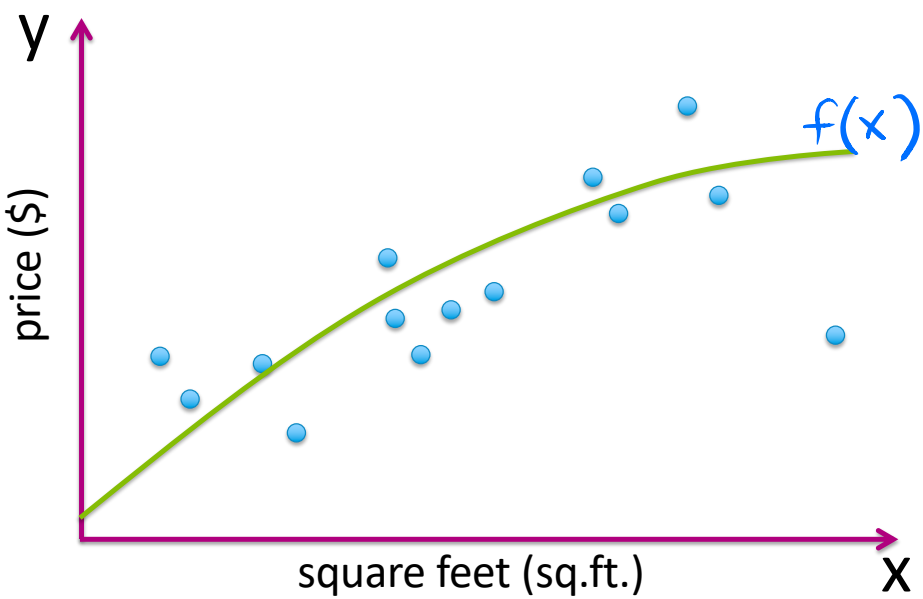
$(x_5 = \text{sq.ft.}, y_5 = \$)$

⋮

Input vs. Output:

- y is the quantity of interest
- assume y can be predicted from x

Model –
How we *assume* the world works



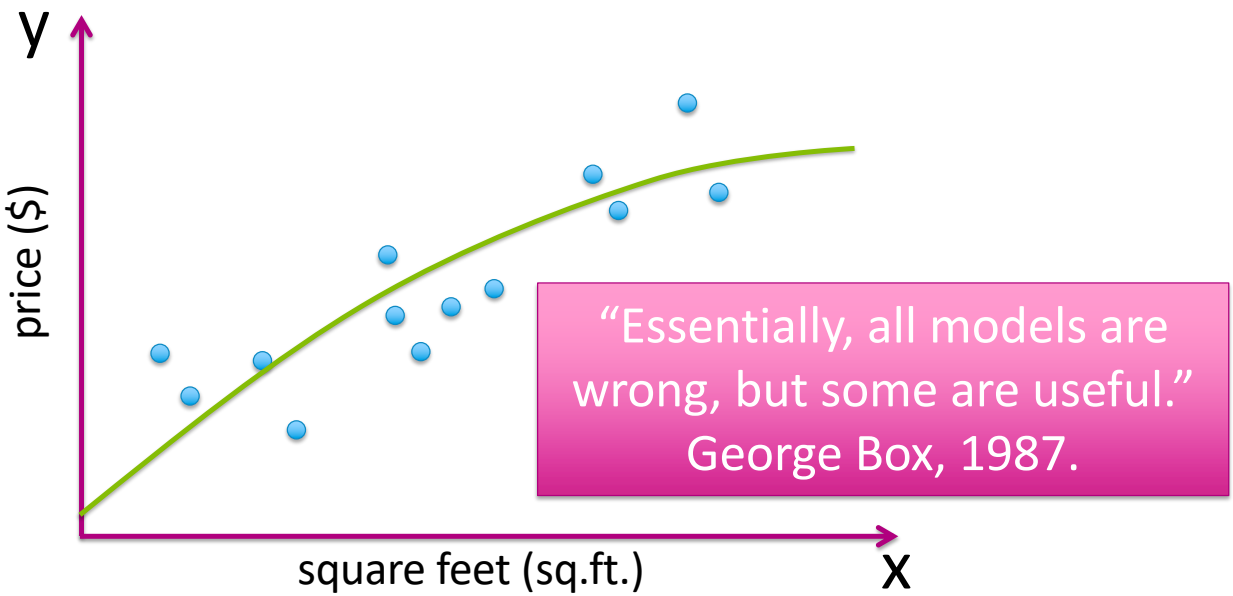
expected relationship between x & y

Regression model:

$$y_i = f(x_i) + \varepsilon_i$$

$$E[\varepsilon_i] = 0$$

Model –
How we *assume* the world works



"Essentially, all models are wrong, but some are useful."
George Box, 1987.

Process

- Chose a model

define a class of functions

$\{ f_w(x) \mid (w_1, \ldots, w_n) \in \mathbb{R}^n \}$

s.t. we believe $\forall x \quad y = f_w(x) + \varepsilon$

- Find the best fit to our data set.

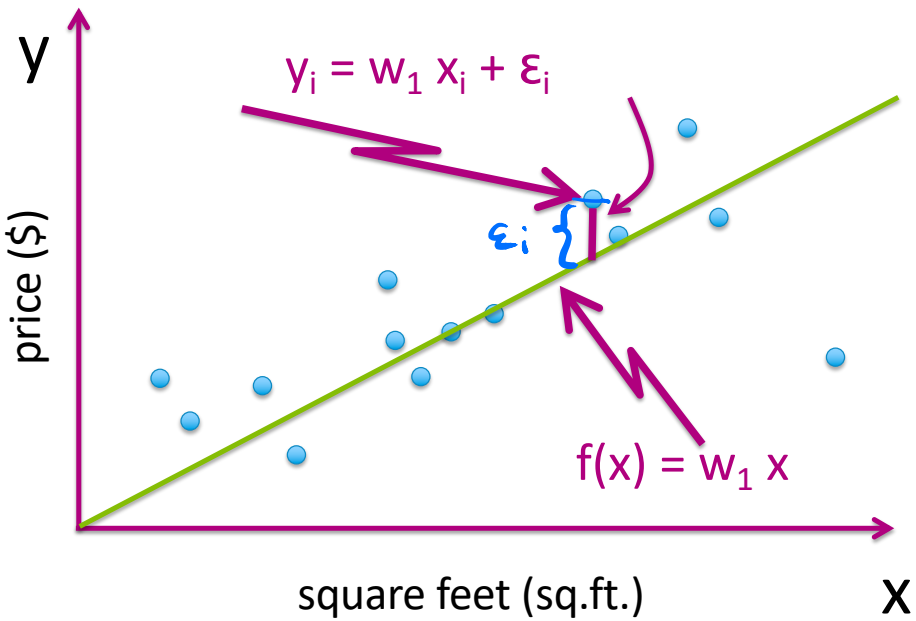
what is best choice for parameters?

$\hat{w}$ best choice

- Use the fitted function to make future predictions.

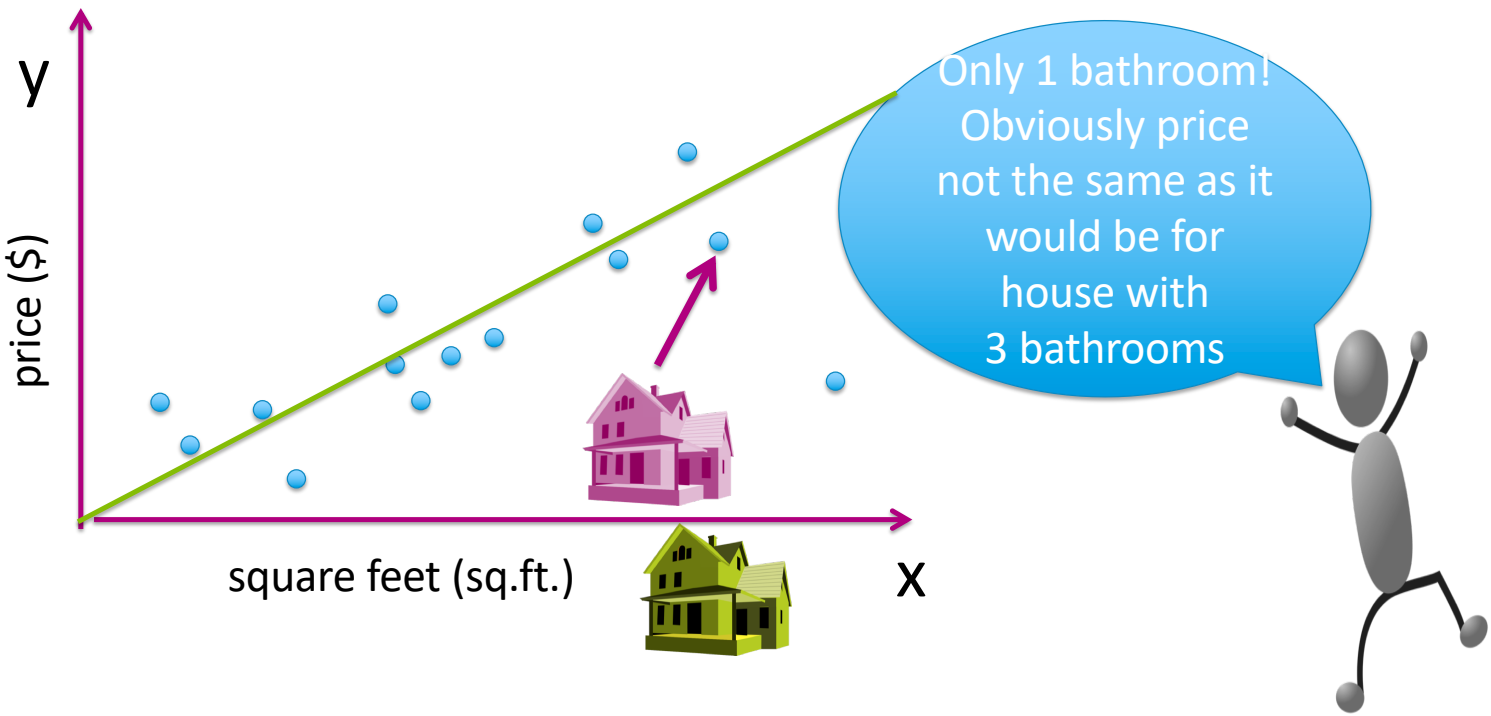
$f_{\hat{w}}(x)$

Simple linear regression model

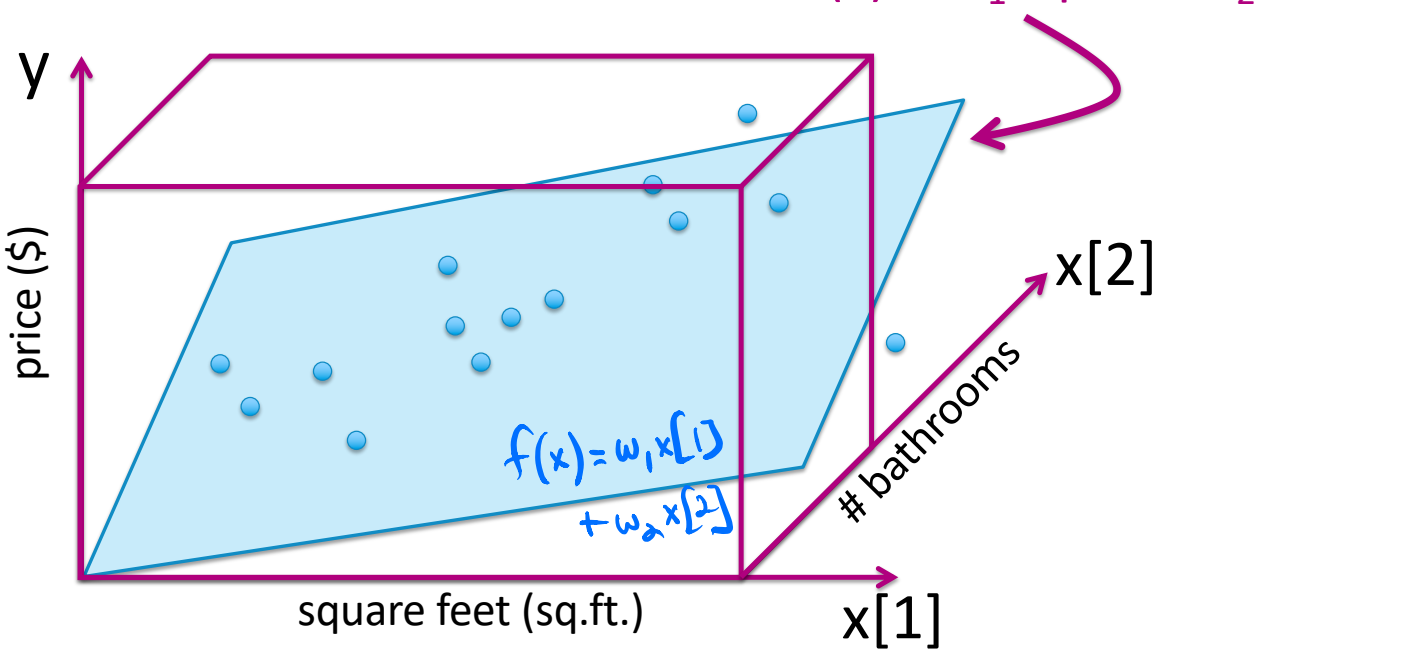


w_1 is the parameter *(regression coefficient) That we want to learn from our data set.*

Predictions just based on house size



Add more inputs



Many possible inputs

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

features.

General notation

d features

Output: y (scalar)

Inputs: **x** = (x[1],x[2],…, x[d]) (d-dim vector)

e.g., x[1] = sq. ft, x[2] = #baths and so on.

Notational conventions:

training set: $\{(\mathbf{x}_i, y_i)\}_{i=1..n}$

$\mathbf{x}_i$ = input of i^{th} data point/observation (*vector*); y_i is output

$x_i[j]$ = j^{th} input of i^{th} data point (*scalar*)

n = number of observations; d = number of input features

Generic linear regression model

Model: Given feature vector $\mathbf{x}_i = (x_i[1], x_i[2], \ldots, x_i[d])$

$$y_i = w_1 x_i[1] + w_2 x_i[2] + \ldots + w_d x_i[d] + \varepsilon_i$$

$$= \sum_{j=1}^{d} w_j x_i[j] + \varepsilon_i \qquad = \mathbf{w}^T \mathbf{x}_i + \varepsilon_i = \mathbf{x}_i^T \mathbf{w} + \varepsilon_i$$

Goal: find best choice for params $w_1, \ldots, w_d$

$$\vec{w} = \begin{pmatrix} w_1 \\ w_2 \\ \\ w_d \end{pmatrix} \qquad x_i = \begin{pmatrix} x_i[1] \\ \vdots \\ x_i[d] \end{pmatrix}$$

feature 1 = x[1] = sq. ft.
feature 2 = x[2] = #bath

…
feature d = x[d] =lot size

$$(w_1 \cdots w_d) \begin{pmatrix} x_i[1] \\ \\ x_i[d] \end{pmatrix}$$

Fitting the linear regression model

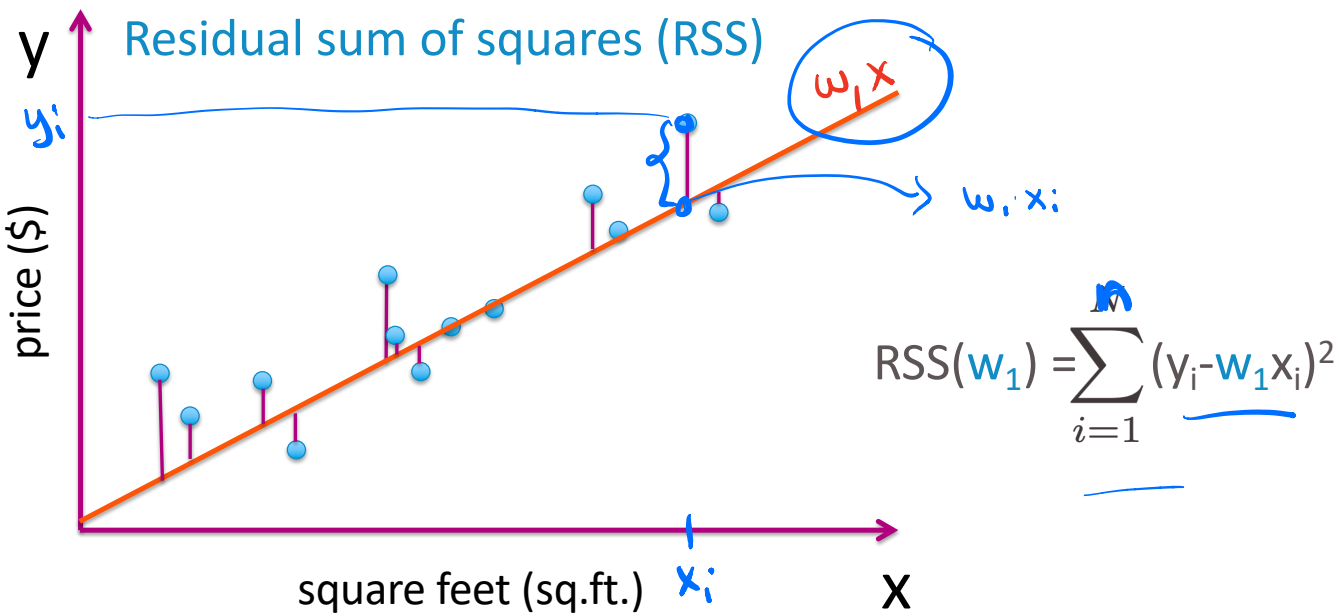
- model
- training set

find best $\vec{w}$

fit $\vec{w}$ parameter set $\vec{w}$ to training set.

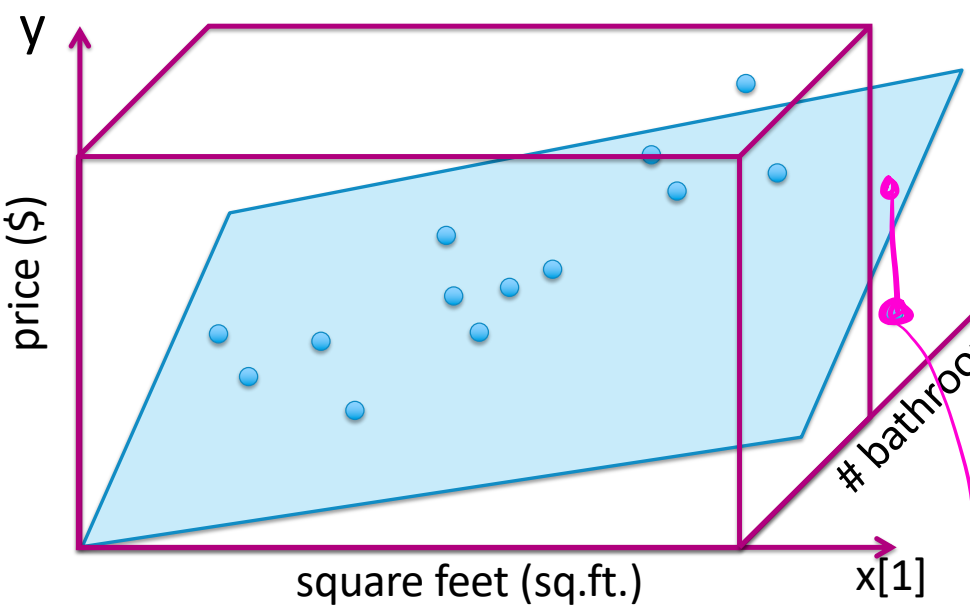
cost($\vec{w}$)
loss($\vec{w}$)

"Cost" of using a given line



$$RSS(w_1) = \sum_{i=1}^{N} (y_i - w_1 x_i)^2$$

RSS for multiple regression



RSS(w)

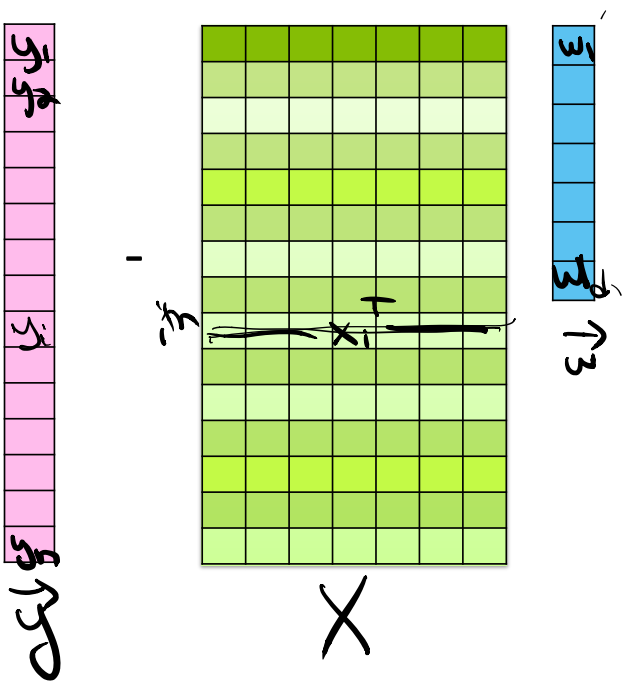
$$= \sum_{i=1}^{n} \left(y_i - \underbrace{x_i^T w}_{x_i[1] w_1 + x_i[2] w_2} \right)^2$$

$(\vec{x_i}, y_i)$

$$RSS(w) = \sum_{i=1}^{n} (y_i - x_i^T w)^2 = (y - Xw)^T (y - Xw) = \|y - Xw\|^2$$

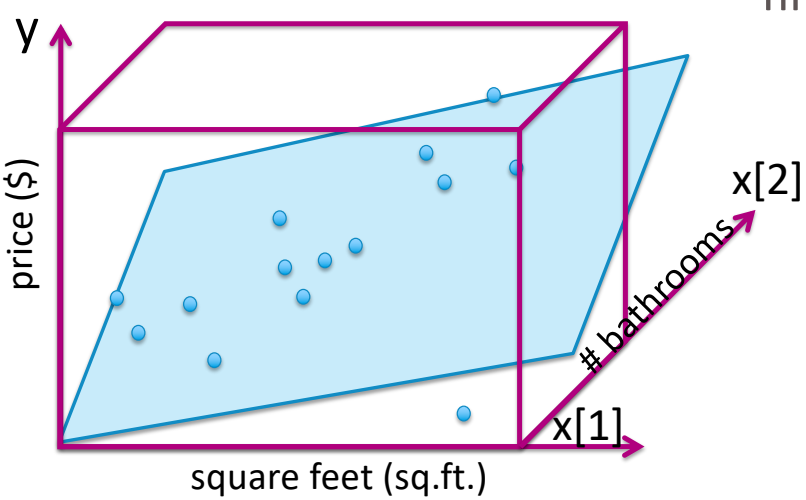
Rewrite in matrix notation

For all observations together



$$\vec{y} - X\vec{w} = \begin{pmatrix} \\ y_i - x_i^T w \\ \\ \end{pmatrix}$$

RSS for multiple regression



Objective: Find the best fit, i.e., find the **w** that minimizes

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

Interlude: Optimization (and convex functions)

Minimizing a univariate function

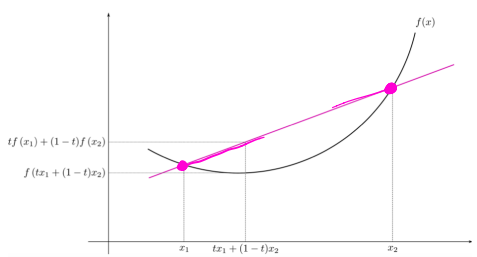
$f(w)$ to $f'(w) = 0$

$\Rightarrow f''(w) > 0$

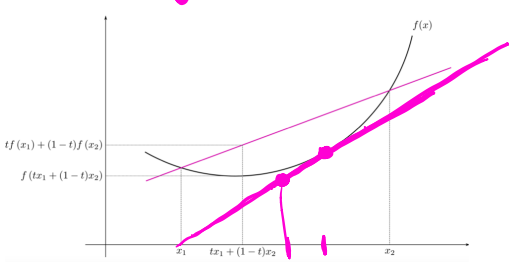
for convex fns if $f'(w) = 0 \Rightarrow w$ is global minimum

Convex functions ← (1) $f''(w) \geq 0$ holds $\forall w$

(2)



(3) $\forall w$ tangent line is global underestimator.



$\ell(w) = f(w_0) + (w - w_0) f'(w_0) \leq f(w)$ $\forall w$

$\ell(w)$: equation of line that is tangent at w_0

0: another means of defining for what it for a fn to be convex.

$f(w_1, \ldots, w_d)$

Minimizing a multivariate function

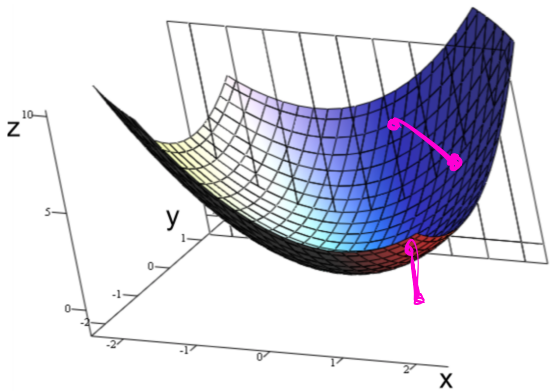
$\nabla f(\vec{w})$ = gradient.

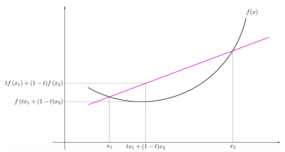
$$= \begin{pmatrix} \vdots \\ \frac{\partial f}{\partial w_i}(\vec{w}) \\ \vdots \end{pmatrix} \longrightarrow$$

$f'(w) = 0 \equiv \nabla f(w) = \vec{0}$

$\nabla f(w) = 0 \Rightarrow$ global min

Convex functions



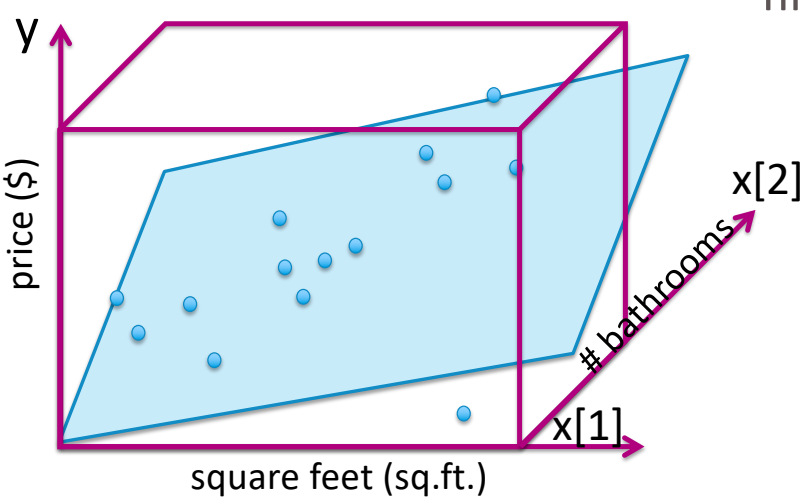


$f(\vec{w}_0) + (w - w_0)^T \nabla f(w_0) \leq f(w)$

$\forall w.$

Back to our specific optimization problem

Two ways to solve our optimization problem



Objective: Find the best fit, i.e., find the **w** that minimizes

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

RSS (**w**) is a convex function of **w**

1. Solve for $\nabla RSS(w) = 0$

Gradient of RSS

$$\vec{w} = \begin{pmatrix} w_1 \\ \vdots \\ w_d \end{pmatrix}$$

$$x_i^T w = \sum_{k=1}^{d} x_i[k]\, w_k$$

$$\nabla RSS(w) = \nabla[(y-Xw)^T(y-Xw)]$$

$$RSS(w) = \sum_i (y_i - x_i^T w)^2$$

$$\Rightarrow \frac{\partial RSS(w)}{\partial w_j} = \sum_i 2(y_i - x_i^T w)(-x_i[j])$$

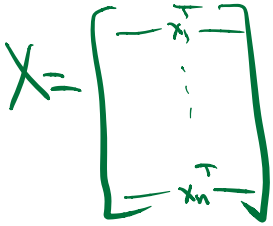
$$\nabla RSS(w) = -2X^T(y - Xw)$$

$y - Xw$

$$-2\begin{bmatrix} | & | & & | \\ x_1 & x_2 & \cdots & x_N \\ | & | & & | \end{bmatrix} \begin{pmatrix} y_1 - x_1^T w \\ \vdots \\ y_N - x_N^T w \end{pmatrix}$$

X^T

$$X = \begin{pmatrix} - & x_1^T & - \\ - & x_2^T & - \\ & \vdots & \end{pmatrix}$$



Closed form solution

$$\nabla RSS(w) = \nabla[(y-Xw)^T(y-Xw)]$$
$$= -2X^T(y-Xw)$$

We want solution to $-2X^T(y-X\hat{w}) = 0$

$X^T y - X^T X \hat{w} = 0$

$\hat{w}$ where $\nabla RSS = 0$.

Solution: ``normal equations'' $X^T X\hat{w} = X^T y$

if $X^T X$ is invertible $\hat{w} = (X^T X)^{-1} X^T y$

$d \times d$

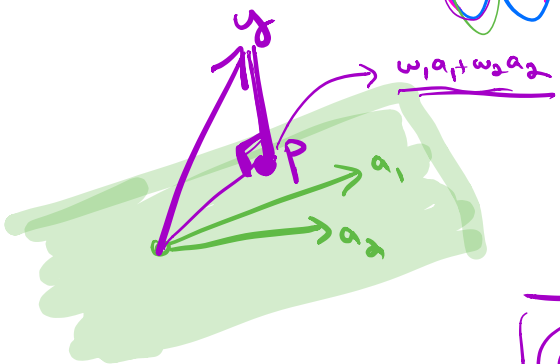
$O(d^3)$

Geometric intuition for solution

Suppose $X =$ 

looking for RSS

w that minimizes $\|y - Xw\|^2$



$y = Xw$

$= w_1 \cdot a_1 + w_2 a_2$

$y - p \perp$ plane

$p = Xw$

$a_1^T (y - Xw) = 0$

$a_2^T (y - Xw) = 0$

$X^T (y - X\hat{w}) = 0$

$\begin{pmatrix} — a_1^T — \\ — a_2^T — \end{pmatrix}$

Xw: predicted sales prices

Again, back to getting closed form solution.

2. Gradient descent

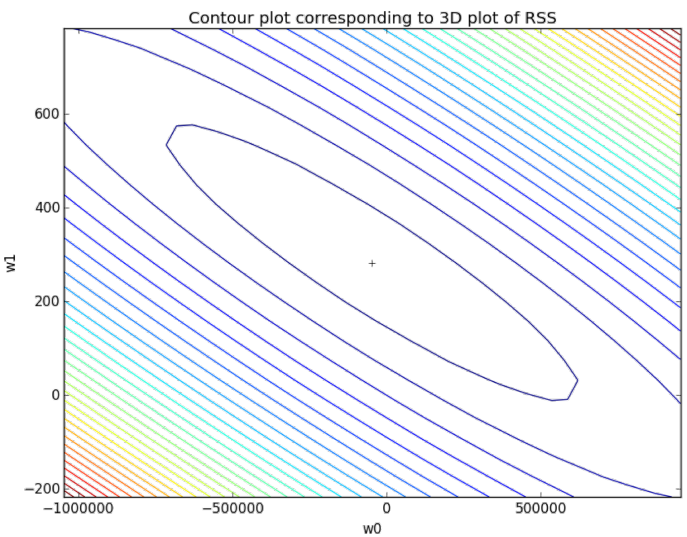
Gradient Descent – univariate case

- Repeatedly move in direction that reduces the value of the function.

Gradient Descent – multivariate case

Gradient descent for linear regression:

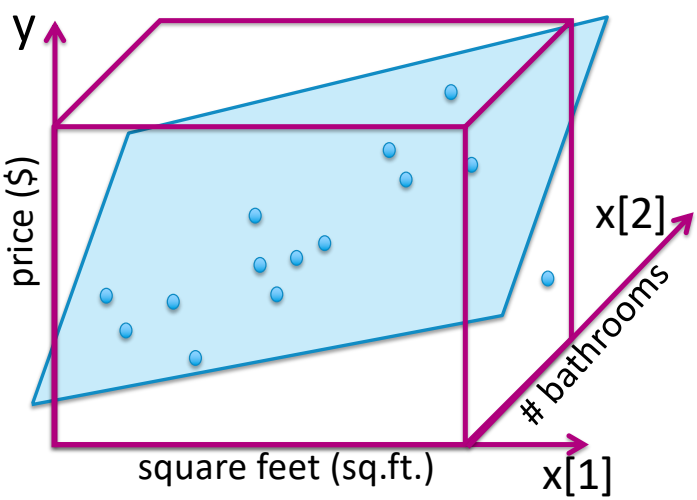
repeatedly move in direction of negative gradient



while not converged

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

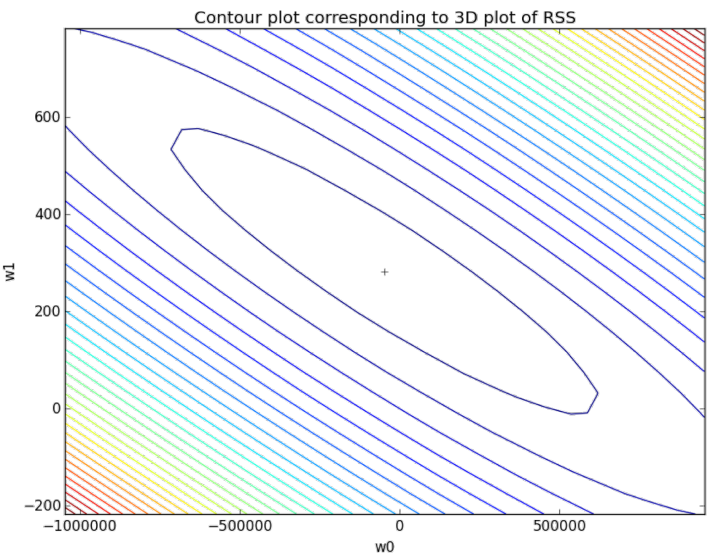
Interpreting elementwise



Update to j^{th} feature weight:

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

Summary of gradient descent for multiple regression



init $w^{(1)}=0$ (or randomly, or smartly), t=1

while $||\nabla RSS(w^{(t)})|| > \varepsilon$

  for j=1,…,d

    partial[j] $=-2\sum_{i=1}^{n} x_i[j](y_i-\hat{y}_i(w^{(t)}))$

    $w_j^{(t+1)} \leftarrow w_j^{(t)} - \eta$ partial[j]

  $t \leftarrow t + 1$

Adding an intercept – “demeaning”

Once we have a fitted function

- We use it to predict the sales price for new houses, by plugging in square footage, number of bathrooms, etc for the new house **x** whose sales price we want to predict.
- Prediction is:

- What if we want to allow for an intercept?

Handling an intercept (constant term)

Two step approach:

1. Show that if $\frac{1}{n}\sum_i \boldsymbol{x}_i = \mathbf{0}$ (*) then solution is simple.

2. Show how to transform, aka ``demean" any linear regression problem so that (*) holds.

1. Show that if $\frac{1}{n}\sum_i \boldsymbol{x}_i = \mathbf{0}$ $(*)$ then solution is simple.

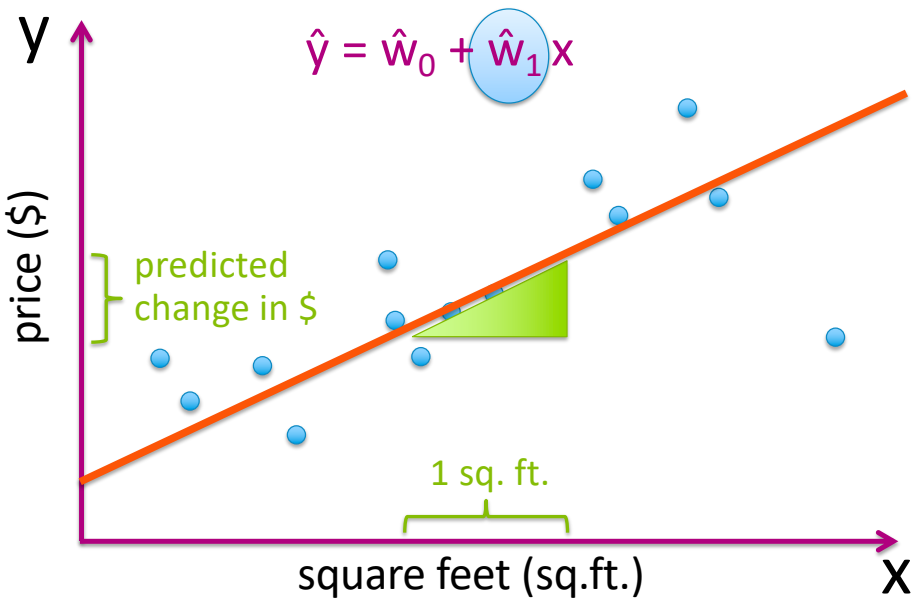
Same as saying that $\mathbf{X}^\top \mathbf{1} = \mathbf{0}$.

2. Show how to transform, aka ``demean" any linear regression problem so that (*) holds.

$$\frac{1}{n}\sum_i \boldsymbol{x}_i = \mathbf{0} \quad (*)$$

Interpreting the fitted function

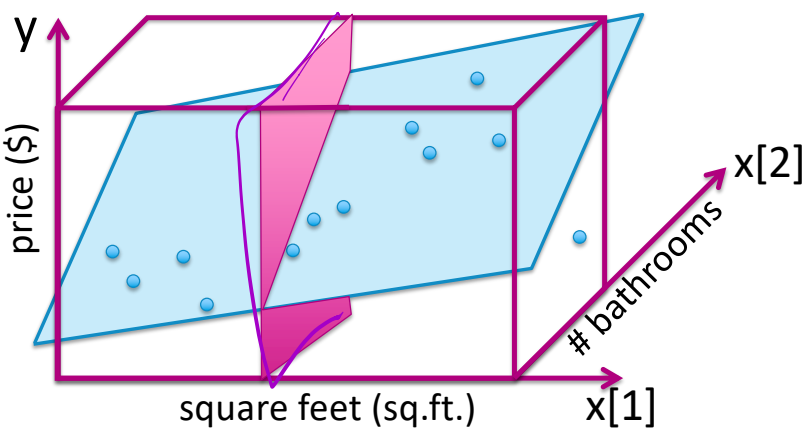
Interpreting the coefficients – Simple linear regression



Interpreting the coefficients – Two linear features

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

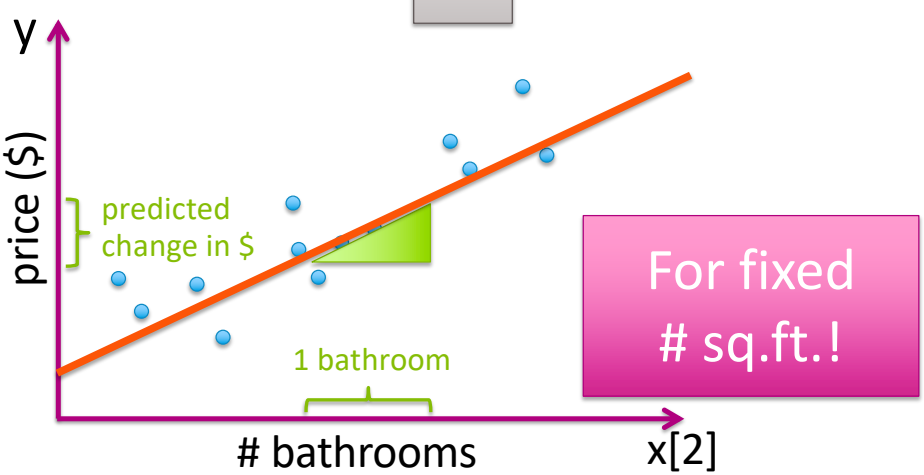
fix



Interpreting the coefficients – Two linear features

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

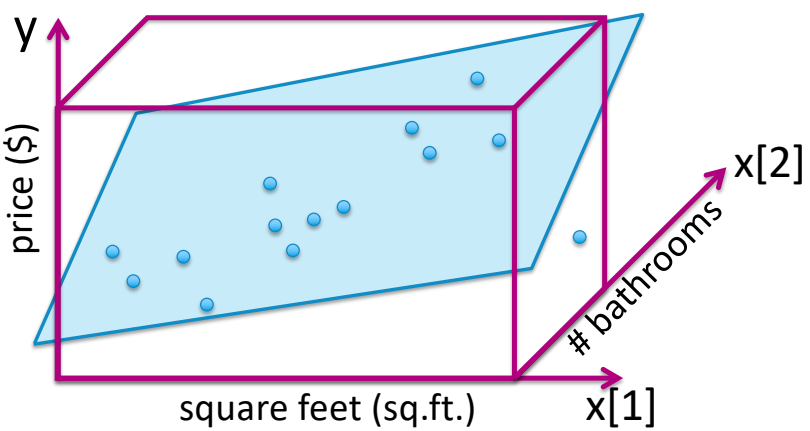
fix



Interpreting the coefficients – Multiple linear features

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

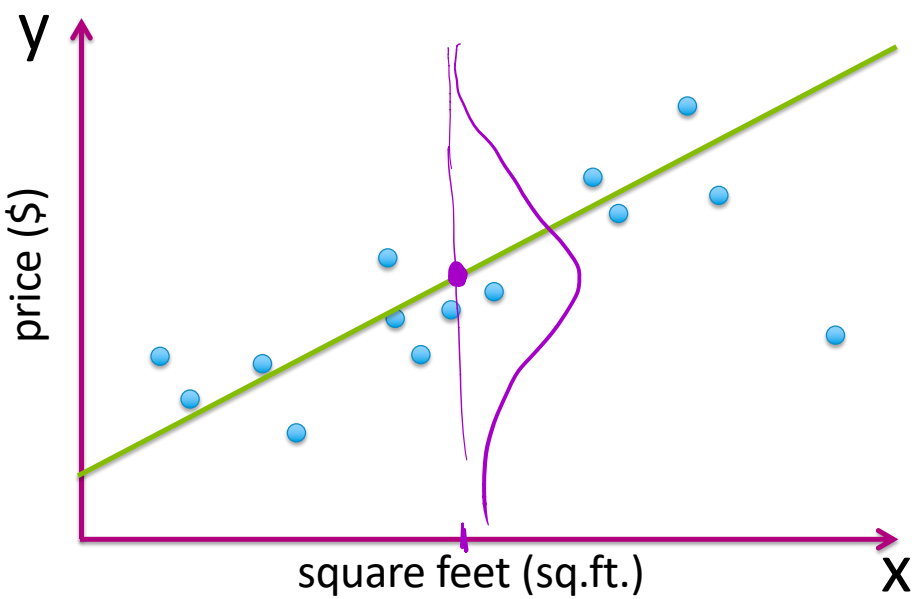
fix fix



Why min RSS?

Assuming Gaussian noise

$y_i = w^T x_i + \varepsilon_i$



Model for ε_i:

$N(0, \sigma^2)$

Implied distribution on y_i:

$N(w^T x_i, \sigma^2)$

Maximum likelihood estimate of params

If $y_i = \mathbf{x}_i^T \mathbf{w} + \varepsilon_i$ is $N(\mathbf{x}^T \mathbf{w}, \sigma^2)$

And we see data $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)$, what is the MLE of $\mathbf{w}$?

$$\log P(y_1 \ldots y_n \mid x_1, \ldots, x_n, w, \sigma^2) = \log \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(y_i - x_i^T w)^2}{2\sigma^2}}$$

$$\log P = \sum_{i=1}^{n} \left[\log\left(\frac{1}{\sqrt{2\pi}\sigma}\right) - \frac{(y_i - x_i^T w)^2}{2\sigma^2} \right]$$

$$\max \text{ log likelihood} \equiv \max \sum_{i=1}^{n} -\frac{(y_i - x_i^T w)^2}{2\sigma^2} \equiv \min \sum_{i=1}^{n} (y_i - x_i^T w)^2$$

Conclusion

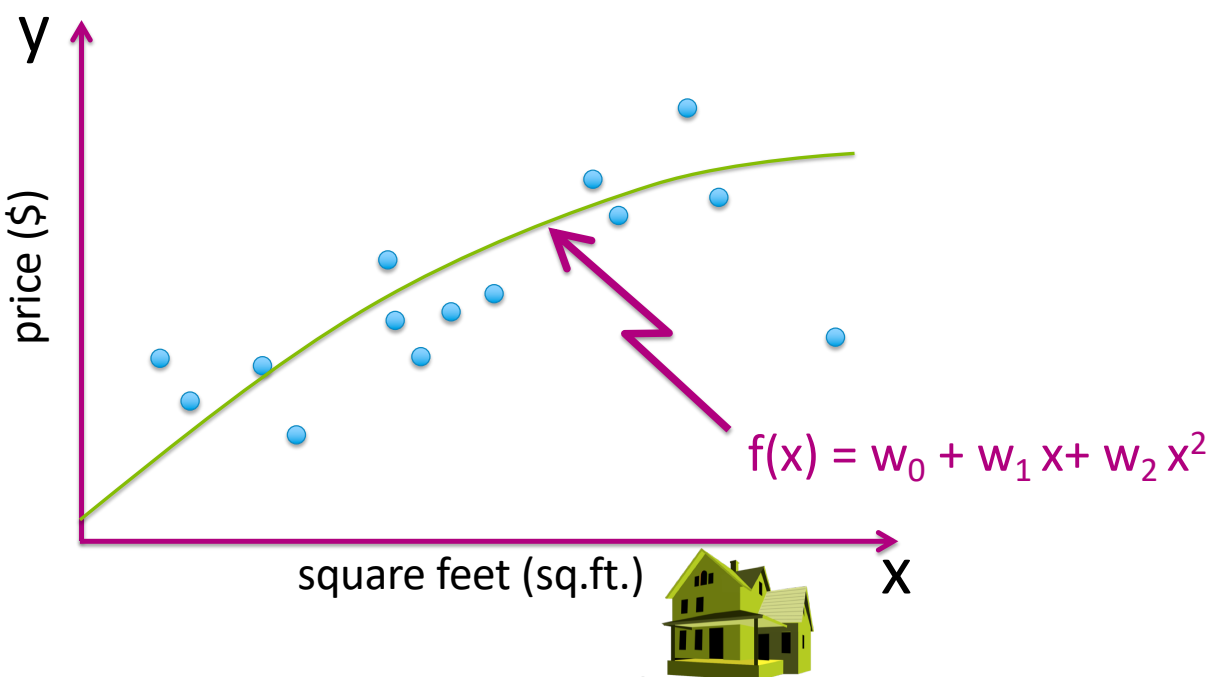
If $y_i = x_i^T \mathbf{w} + \varepsilon_i$ is $N(x^T w, \sigma^2)$

Then choosing w to maximize log-likelihood

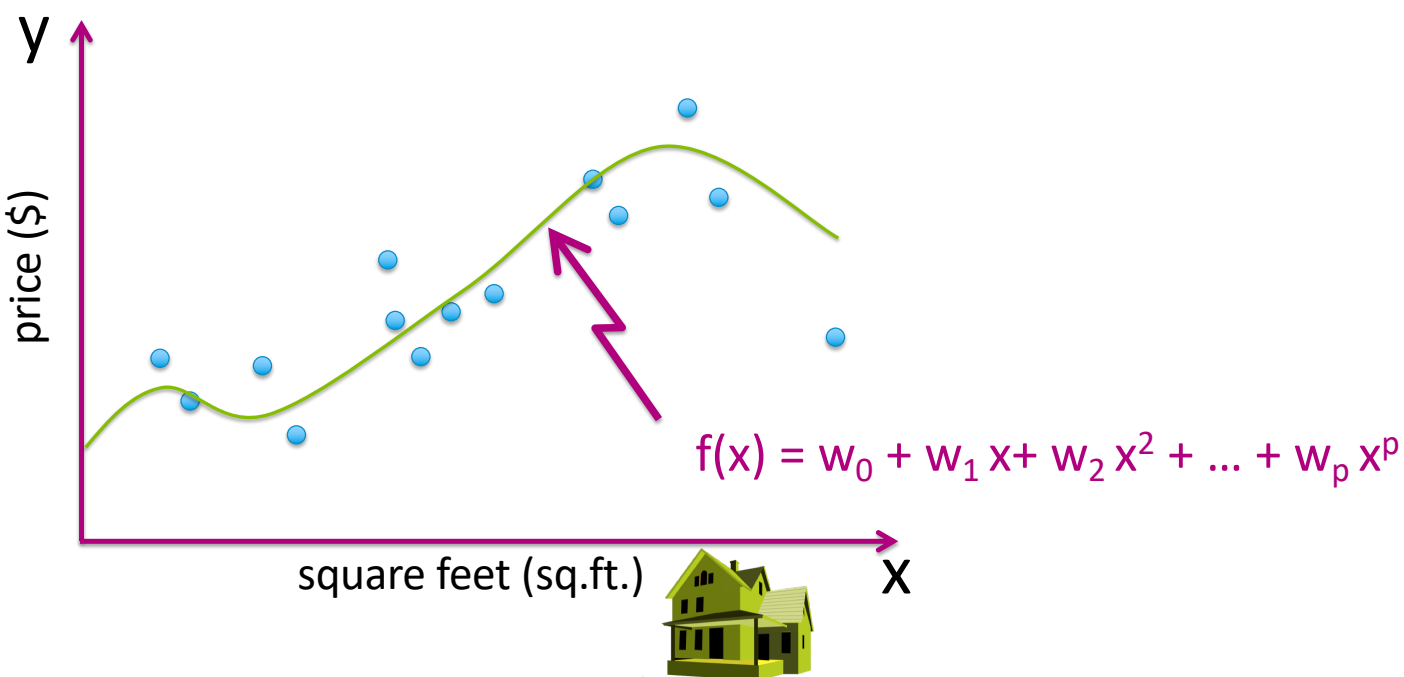
is same as choosing w to minimize RSS!

Feature maps, polynomial regression and basis expansion

What about a quadratic function?



Even higher order polynomial



Polynomial Regression

- Start with single input feature x (e.g. square footage of house)

and training set: $\{(x_i, y_i)\}_{i=1..n}$

- Define feature map that transforms each x_i to higher dimensional feature vector $h(x_i)$.

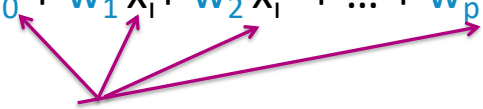
Example: x_i

$h_0(x_i)$	$h_1(x_i)$	$h_2(x_i)$	$h_3(x_i)$	$h_4(x_i)$	$h_5(x_i)$
1	x_i	x_i^2	x_i^3	x_i^4	x_i^5

Polynomial regression

Model:

$y_i = w_0 + w_1 x_i + w_2 x_i^2 + \ldots + w_p x_i^p + \varepsilon_i$



treat transformed inputs 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

Why might we want to use polynomial regression?

- Taylor Series!

More generally

- Start with input features $\mathbf{x} = (x[1], x[2], \ldots, x[d])$

and training set: $\{(\mathbf{x}_i, y_i)\}_{i=1..n}$

- Define feature map that transforms each $\mathbf{x}_i$ to higher dimensional feature vector $h(\mathbf{x}_i)$.

Example: $x_i[1]$ $x_i[2]$ $x_i[3]$

$h_1(\mathbf{x})$	$h_2(\mathbf{x}_i)$	$h_3(\mathbf{x}_i)$	$h_4(\mathbf{x}_i)$,	$h_5(\mathbf{x}_i)$	$h_6(\mathbf{x}_i)$	$h_7(\mathbf{x}_i)$
1	$x_i[1]$	$x_i[1]^2$	$x_i[1]x_i[2]$	$x_i[2]$	$x_i[2]^2$	$\cos(\pi\, x_i[3]/6)$

General notation

Output: y ← scalar

Inputs: **x** = (x[1],x[2],…, x[d])

↖ d-dim vector

Notational conventions:

- $\mathbf{x}_i$ = input of i^{th} data point (*vector*)
- $x_i[j]$ = j^{th} input of i^{th} data point (*scalar*)
- $h(\mathbf{x}) = (h_1(\mathbf{x}), h_2(\mathbf{x}), \ldots, h_d(\mathbf{x}))$ feature map applied to input x (*vector*)
- $h_j(\mathbf{x})$ = j^{th} feature associated with input x (*scalar*) (j^{th} basis function)

To fit these more general functions

- Start with input features $\mathbf{x}$ = (x[1],x[2],…, x[d])

and training set: $\{(\mathbf{x}_i, y_i)\}_{i=1..n}$

- Define feature map that transforms each $\mathbf{x}_i$ to higher dimensional feature vector $h(\mathbf{x}_i)$.
- Model: $y_i = \sum_{j=1}^{p} w_j h_j(\mathbf{x}_i) + \varepsilon_i$
- Find $\hat{w}$ that minimizes RSS = $\sum_{i=1}^{n} (y_i - \sum_{j=1}^{p} w_j h_j(\mathbf{x}_i))^2$

Recap of concepts

What you can do now…

- Describe linear regression (and feature maps)
- Write a regression model using multiple inputs or features thereof.
- 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