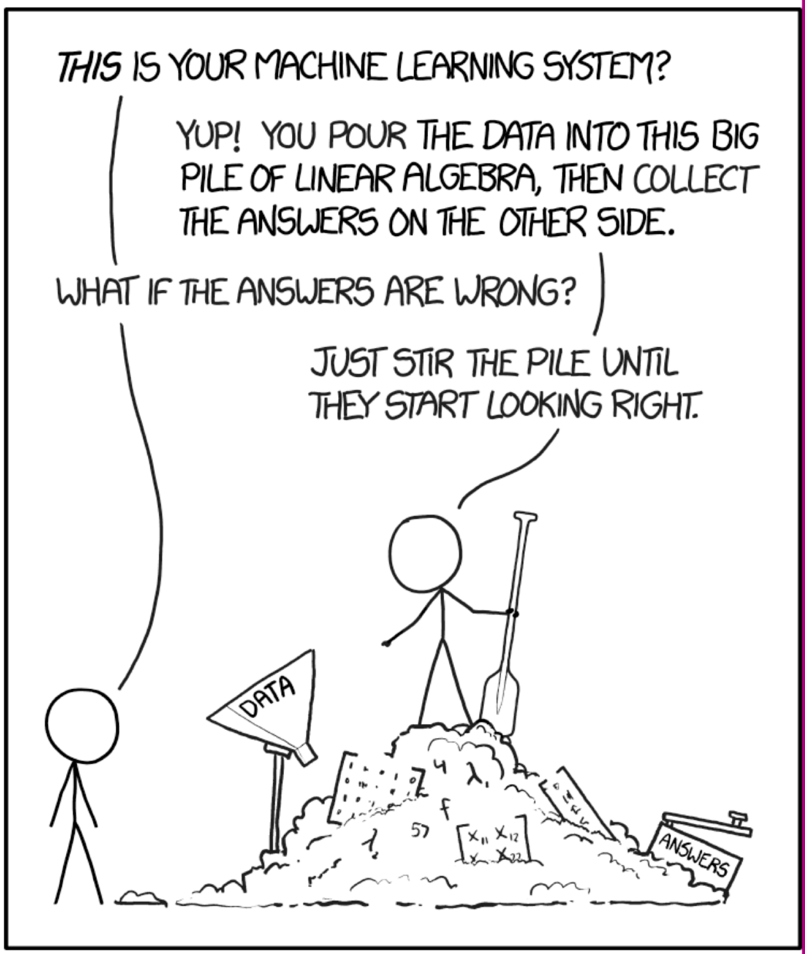
# Linear Regression: Model and Algorithms (part 2)

CSE 446

Most slides by Emily Fox
Presented by Anna Karlin
April 8, 2019



THIS IS YOUR MACHINE LEARNING SYSTEM?
YUP! YOU POUR THE DATA INTO THIS BIG PILE OF LINEAR ALGEBRA, THEN COLLECT THE ANSWERS ON THE OTHER SIDE.
WHAT IF THE ANSWERS ARE WRONG?
JUST STIR THE PILE UNTIL THEY START LOOKING RIGHT.
DATA
ANSWERS


XKCD

# Linear regression: a supervised learning problem

**Goal:** to predict some output from some inputs using labelled examples. Example: house sales price from square footage

**Supervised learning:** Problem of learning a function that maps inputs to outputs based on labelled examples.

**Regression:** When the labels are real numbers

# Linear regression: a supervised learning problem

**Goal:** to predict some output from some inputs/features. Example: house sales price from square footage

**Step 1:** Define set up and get data

- a model for how the output y depends on the inputs **x**.
- We assumed that y is a linear function of features + noise.

$$y = \mathbf{w}^T \mathbf{x} + \varepsilon$$

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

# Linear regression: a supervised learning problem

**Goal:** to predict some output from some inputs/features.

**Step 1:** Define set up ($y = \mathbf{w}^T \mathbf{x} + \varepsilon$) and get data $\{(\mathbf{x}_i, y_i)\}_{i=1..n}$

**Step 2:** find the parameters $\mathbf{w}$ that minimize the "loss/cost" on the training set.

- Our **loss function** was residual sum of squares (RSS)
- Find $\hat{w}$ that minimizes RSS = $\sum_{i=1}^{n} (y_i - \sum_{j=1}^{d} w_j\, x_i[j])^2 = (\mathbf{y}-X\mathbf{w})^T(\mathbf{y}-X\mathbf{w})$
- Found solution by solving for gradient of $(\mathbf{y}-X\mathbf{w})^T(\mathbf{y}-X\mathbf{w}) = \mathbf{0}$

Solution: $X^TX\hat{w} = X^Ty$ If $X^TX$ is invertible, could write $\hat{w} = (X^TX)^{-1}X^T\mathbf{y}$

# Linear regression: a supervised learning problem

**Goal:** to predict some output from some inputs/features.

**Step 1:** Define set up ($y = \mathbf{w}^T \mathbf{x} + \varepsilon$) and get data $\{(\mathbf{x}_i, y_i)\}_{i=1..n}$

**Step 2:** find the parameters $\hat{\mathbf{w}}$ that minimizes RSS = $(\mathbf{y}-X\mathbf{w})^T(\mathbf{y}-X\mathbf{w})$

**Step 3:** Use $\hat{\mathbf{w}}$ to make predictions.

Given $\mathbf{x}$, predict output: $\hat{\mathbf{w}}^T \mathbf{x}$

**Plan for today:**

- Gradient descent
- Handling an intercept
- More features/more complex models
- How well does it work?

# Linear regression: a supervised learning problem

**Goal:** to predict some output from some inputs/features. Example: house sales price from square footage

**Step 1:** Define set up and get data

- a model for how the output y depends on the inputs x.
- We assumed that y is a linear function of features + noise.

  $y = \mathbf{w}^T \mathbf{x} + \varepsilon$

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

**Step 2:** find the parameters **w** that minimize the "loss/cost" on the training set.

- Our **loss function** was residual sum of squares (RSS)
- Find $\hat{w}$ that minimizes $\text{RSS} = \sum_{i=1}^{n} (y_i - \sum_{j=1}^{d} \mathbf{x}_i{}^T \mathbf{w}_j)^2 = (\mathbf{y}-X\mathbf{w})^T(\mathbf{y}-X\mathbf{w})$
- Found solution by solving for gradient of $(\mathbf{y}-X\mathbf{w})^T(\mathbf{y}-X\mathbf{w}) = \mathbf{0}$

  Solution: $X^T X\hat{w} = X^T y$ If $X^T X$ is invertible, could write $\hat{w} = (X^T X)X^t\mathbf{y}$

**Step 3:** Use $\hat{w}$ to make predictions, Given **x**, predict output: $\mathbf{w}^T \mathbf{x}$

## Fitting the linear regression model

## Gradient descent

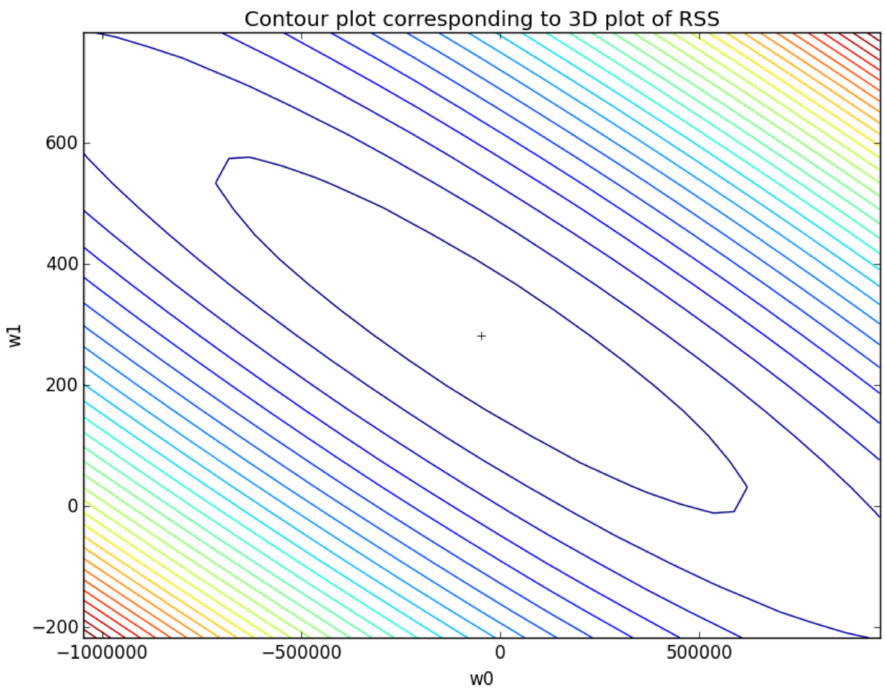
# Gradient Descent – univariate case

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

# Gradient Descent – multivariate case

# Gradient descent for linear regression:

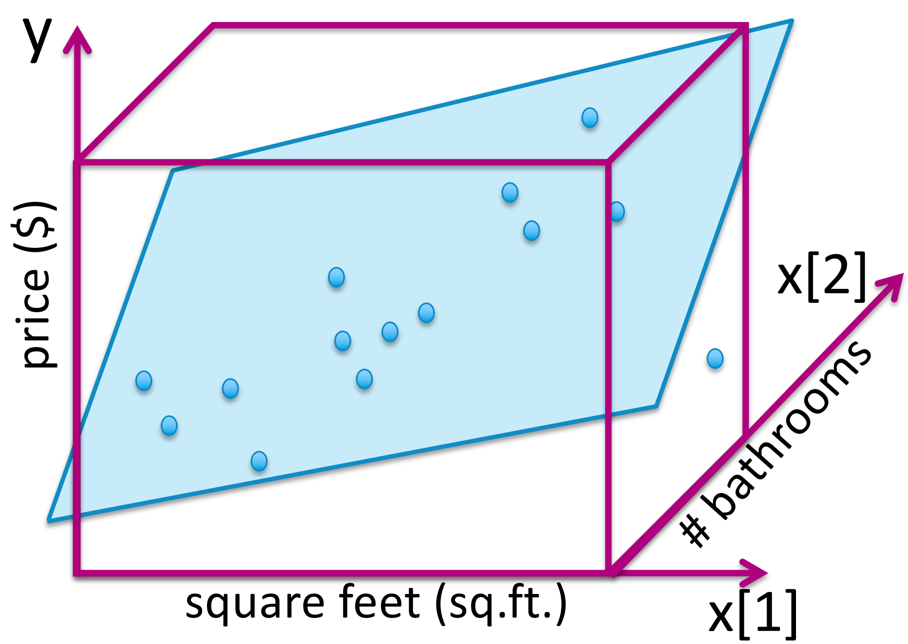
## repeatedly move in direction of negative gradient




while not converged

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

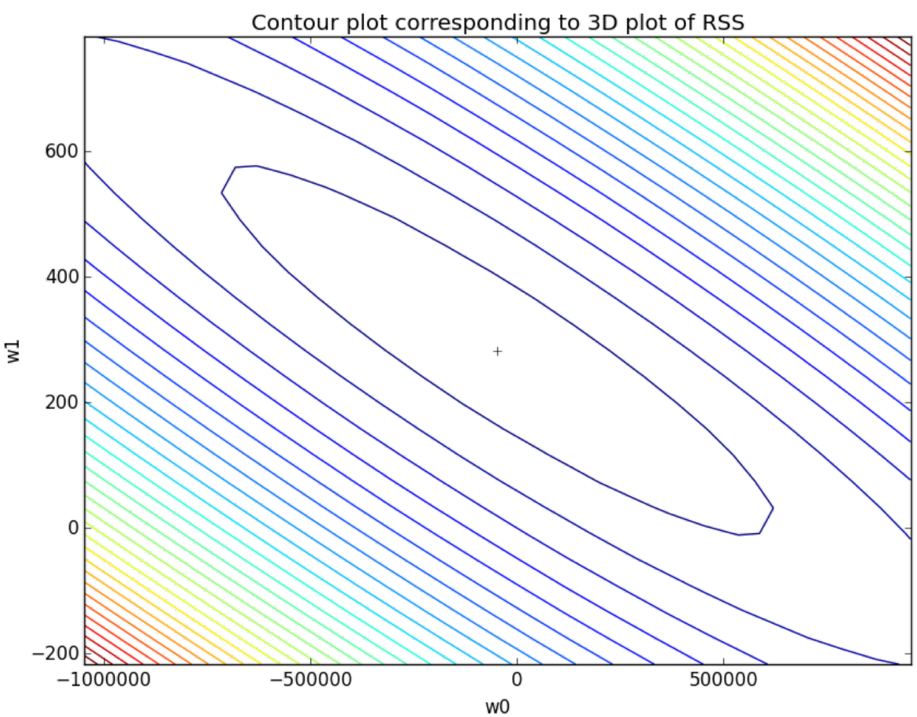
# Interpreting elementwise




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

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

  for j=1,…,d

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

    $w_j^{(t+1)} \leftarrow w_j^{(t)} - \eta$ 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?

Assume that $y = \mathbf{w}^T \mathbf{x} + b + \varepsilon$

Find $\hat{w}, b$ that minimize $\quad RSS = \sum_{i=1}^{n} (y_i - \sum_{j=1}^{d} w_j x_i[j] - b)^2$

$$= (\mathbf{y} - X\mathbf{w} - b\mathbf{1})^T(\mathbf{y} - X\mathbf{w} - b\mathbf{1})$$

# Handling an intercept (constant term)

Assume that $y = \mathbf{w}^T \mathbf{x} + b + \varepsilon$

Find $\hat{w}$, $b$ that minimize $\quad \text{RSS} = \sum_{i=1}^{n} (y_i - \sum_{j=1}^{d} w_j x_i[j] - b)^2$

$$= (\mathbf{y} - X\mathbf{w} - b\mathbf{1})^T(\mathbf{y} - X\mathbf{w} - b\mathbf{1})$$

Two step approach:

1. Show that if $\frac{1}{n}\sum_i \boldsymbol{x}_i = \mathbf{0} \quad (*)$ 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 $X^T\mathbf{1} = \mathbf{0}$.

Find $\hat{w}$, $b$ that minimize

RSS = $\sum_{i=1}^{n} (y_i - \sum_{j=1}^{d} w_j x_i[j] - b)^2$

= $(\mathbf{y} - X\mathbf{w} - b\mathbf{1})^T(\mathbf{y} - X\mathbf{w} - b\mathbf{1})$

partial[$w_j$] $= -2 \sum_{i=1}^{n} x_i[j](y_i - \mathbf{x}_i^T\mathbf{w} - b)$

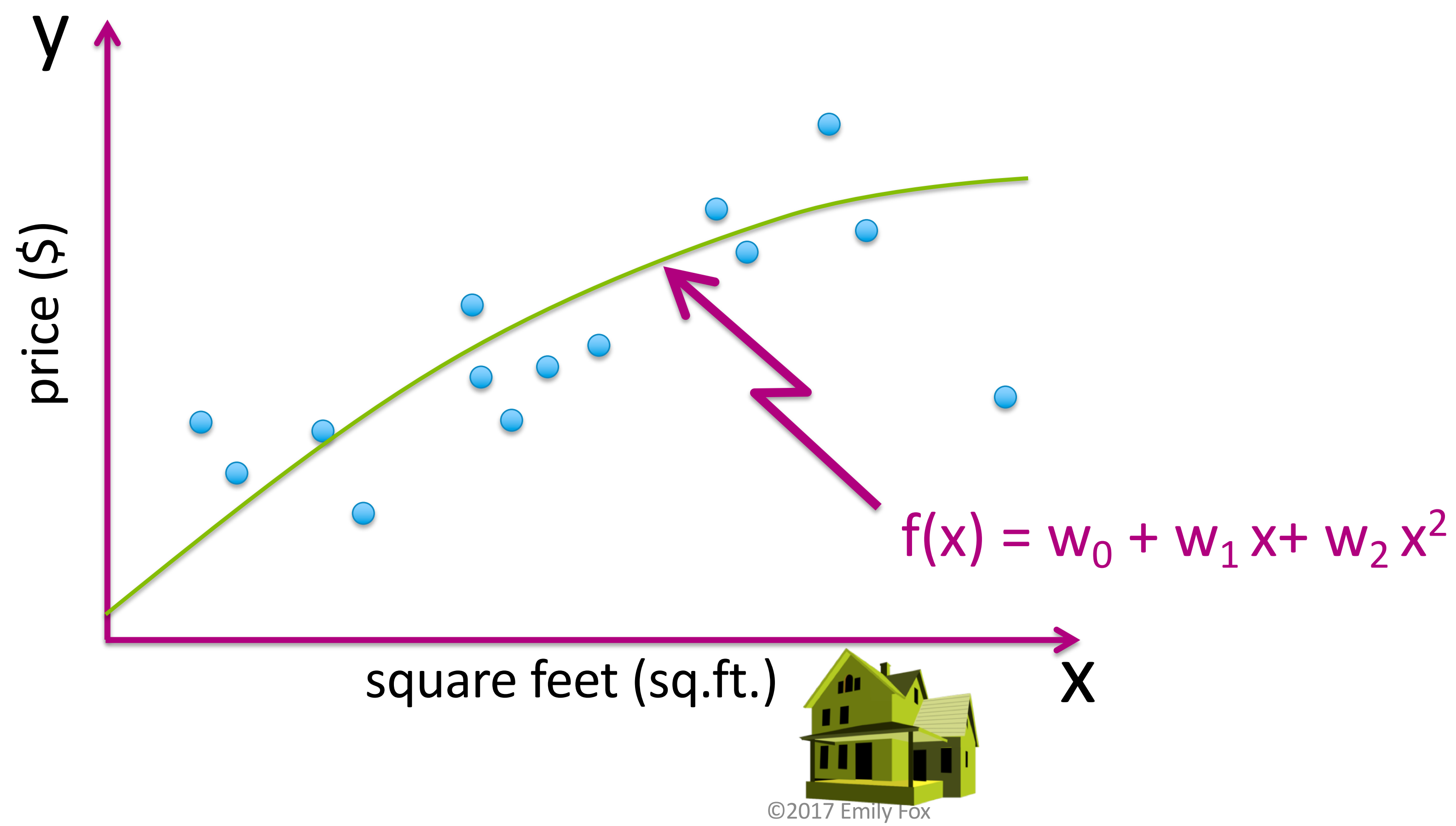
partial[$b$] $= -2 \sum_{i=1}^{n} (y_i - \mathbf{x}_i^T\mathbf{w} - b)$

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

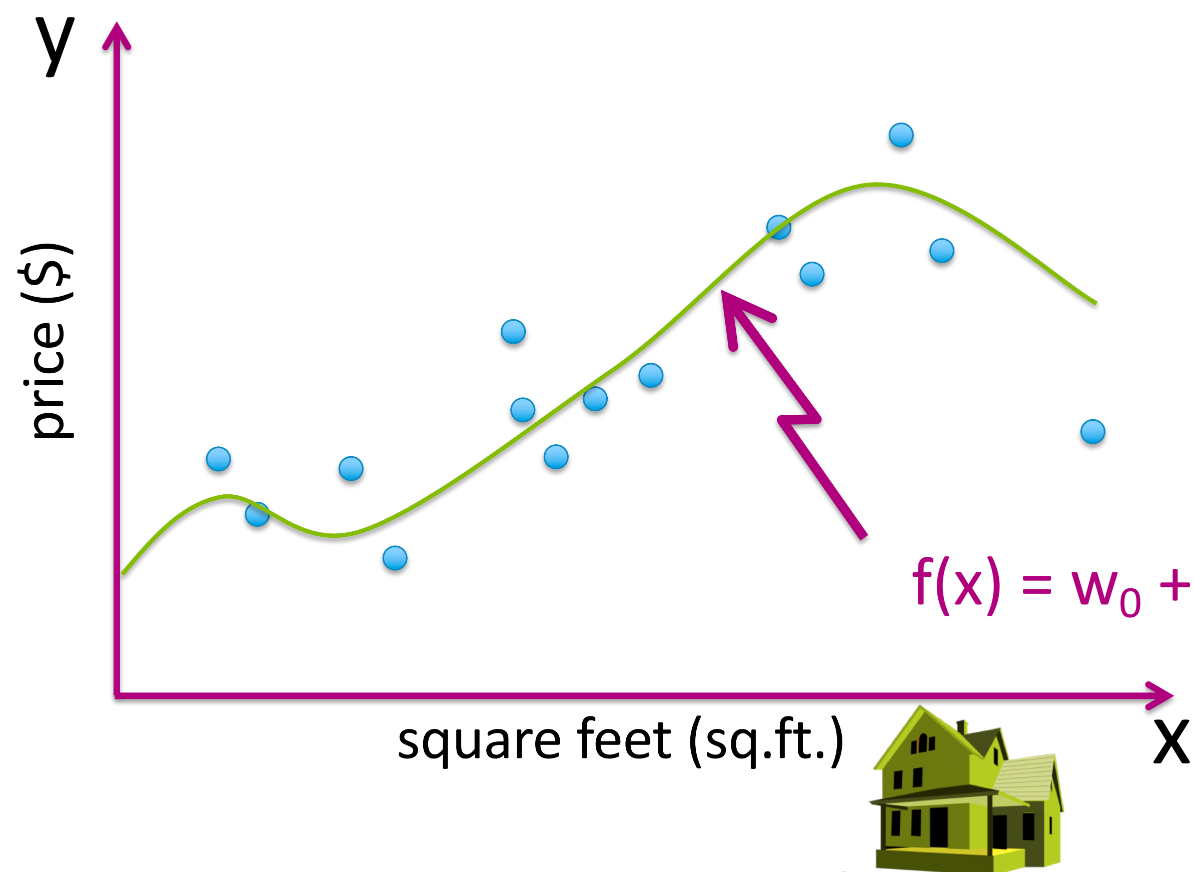
$$\frac{1}{n}\sum_i \boldsymbol{x}_i = \mathbf{0} \quad (*)$$

# More features, more complex models

# What about a quadratic function?



# Even higher order polynomial



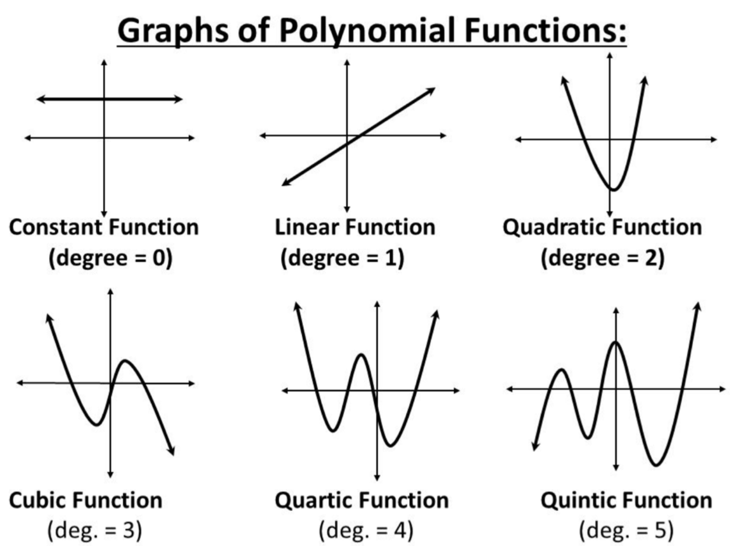

$$f(x) = w_0 + w_1 x + w_2 x^2 + \ldots + w_p x^p$$

# Polynomial regression (single input)

**Goal:** to predict some output from some inputs/features.

**Step 1:** Assume that y (sales price) is a polynomial function of feature (square footage)+ noise.

$y_i = \sum_{j=0}^{p} w_j x_i^j + \varepsilon$ A training set (labelled examples):



# Polynomial regression

**Goal:** to predict some output from some inputs/features.

**Step 1:** $y_i = \sum_{j=0}^{p} w_j \, x_i^j + \varepsilon$ A training set (labelled examples):

**Step 2:** find params **w** that minimize the "loss/cost" on training set $\{(x_i, y_i)\}_{i=1..n}$

- **Loss function** is residual sum of squares (RSS)
- Find $\hat{\mathbf{w}}$ that minimizes RSS = $\sum_{i=1}^{n} (y_i - \sum_{j=0}^{p-1} w_j \, x_i^j)^2$

# Polynomial regression

**Goal:** to predict some output from some inputs/features.

**Step 1:** $y_i = \sum_{j=0}^{p} w_j x_i^j + \varepsilon$ A training set (labelled examples):

**Step 2:** find params **w** that minimize the "loss/cost" on training set $\{(x_i, y_i)\}_{i=1..n}$

- Find $\hat{w}$ that minimizes RSS = $\sum_{i=1}^{n} (y_i - \sum_{j=0}^{p-1} w_j x_i^j)^2$

- **Just as easy to solve!** Just think of $x_i^j$ as one of p features associated with the $i^{th}$ observation.
- Instead of single input $x_i$, define features **h(**x**)** = $(1, x, x^2 ..., x^p)$

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

# Polynomial regression

**Step 1:** $y_i = \sum_{j=0}^{p} w_j x_i^j + \varepsilon$

**Step 2:** find the parameters **w** that minimize the "loss/cost" on the training set.

- Find $\hat{\mathbf{w}}$ that minimizes RSS = $\sum_{i=1}^{n} (y_i - \sum_{j=0}^{p} \mathbf{w_j} x_i^j)^2 = (\mathbf{y}-H\mathbf{w})^T(\mathbf{y}-H\mathbf{w})$
- Find solution by solving for gradient of $(\mathbf{y}-H\mathbf{w})^T(\mathbf{y}-H\mathbf{w}) = \mathbf{0}$

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

# Polynomial regression

**Step 1:** $y_i = \sum_{j=0}^{p} w_j x_i^j + \varepsilon$

**Step 2:** find the parameters **w** that minimize the "loss/cost" on the training set.

- Find $\hat{\mathbf{w}}$ that minimizes RSS = $\sum_{i=1}^{n} (y_i - \sum_{j=0}^{p} \mathbf{w}_j x_i^j)^2 = (\mathbf{y}-H\mathbf{w})^T(\mathbf{y}-H\mathbf{w})$
- Find solution by solving for gradient of $(\mathbf{y}-H\mathbf{w})^T(\mathbf{y}-H\mathbf{w}) = \mathbf{0}$

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

**Step 3:** Use $\hat{\mathbf{w}} = (\hat{w}_0, \hat{w}_1 \ldots, \hat{w}_p)$ to make predictions. Given x, let

$\mathbf{h}(x) = (1, x, x^2 \ldots, x^p)$ and predict output:

$$\mathbf{h}(x)^T \hat{\mathbf{w}} = \hat{w}_0 + \hat{w}_1 x + \ldots + \hat{w}_p x^p$$

# 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)

*feature 2* = x

*feature 3* = $x^2$

...

*feature p+1* = $x^p$

*parameter* 1 = $w_0$

*parameter* 2 = $w_1$

*parameter* 3 = $w_2$

...

*parameter* p+1 = $w_p$

# Why might we want to use polynomial regression?

- Taylor Series!

# More generally

- Start with set of inputs for each observation $\mathbf{x} = (x[1],x[2],\ldots, x[d])$

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

- Define feature map that transforms each input vector $\mathbf{x}_i$ 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

scalar

Output: y

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

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*)

$\mathbf{h}(\mathbf{x}) = (h_1(\mathbf{x}), h_2(\mathbf{x}), \ldots, h_p(\mathbf{x}))$ feature map applied to input $\mathbf{x}$ (*vector*)

$h_j(\mathbf{x})$ = $j^{th}$ feature associated with input x (*scalar*) ($j^{th}$ basis function)

H = n by p matrix whose ith row is $\mathbf{h}(\mathbf{x}_i)$

# To fit these more general functions

- 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 $\mathbf{h}(\mathbf{x}_i)$.
- Model: $y_i = \sum_{j=1}^{p} w_j h_j(\mathbf{x}_i) + \varepsilon_i$
- Find $\hat{\mathbf{w}}$ that minimizes RSS $= \sum_{i=1}^{n} (y_i - \sum_{j=1}^{p} w_j h_j(\mathbf{x}_i))^2$

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

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

# 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