# CPSC 340:
# Machine Learning and Data Mining

Non-Linear Regression

# Admin

- Assignment 1 grades are out.
- Assignment 2 is due Sunday.
- Assignment 3 will be out by early next week.
  - Will be due before the break
- Midterm is after the break (March 1 in class)
- Tutorial next week: practice problems for hw3

# Last Time: Linear Regression

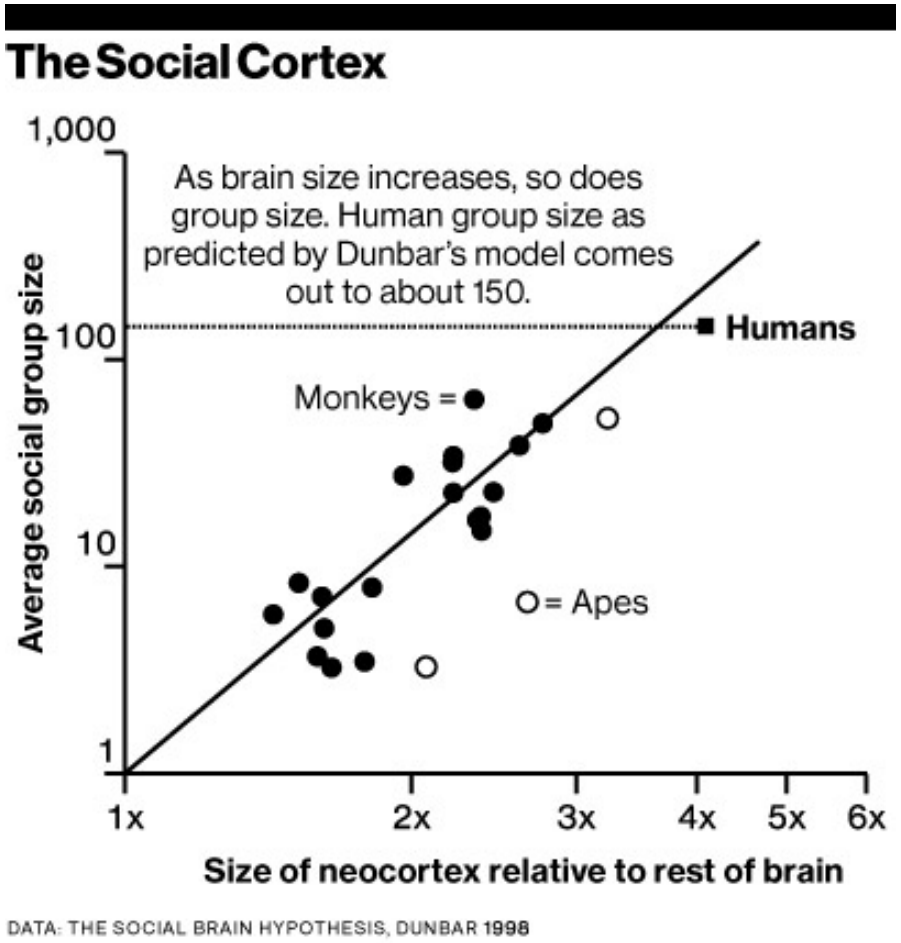
- We discussed linear models:

$$y_i = w_1 x_{i1} + w_2 x_{i2} + \cdots + w_d x_{id}$$
$$= \sum_{j=1}^{d} w_j x_{ij} = w^T x_i$$

- "Multiply feature $x_{ij}$ by weight $w_j$, add them to get $y_i$".
- We discussed squared error function:

$$f(w) = \frac{1}{2} \sum_{i=1}^{n} (w^T x_i - y_i)^2$$

Predicted value ↑ ↓ True value




To predict on test case $\hat{x}_i$ use $\hat{y}_i = w^T \hat{x}_i$

http://www.bloomberg.com/news/articles/2013-01-10/the-dunbar-number-from-the-guru-of-social-networks

# Last Time: Supervised Learning Notation

- We're treating 'w', 'y', and each $x_i$ as column-vectors:

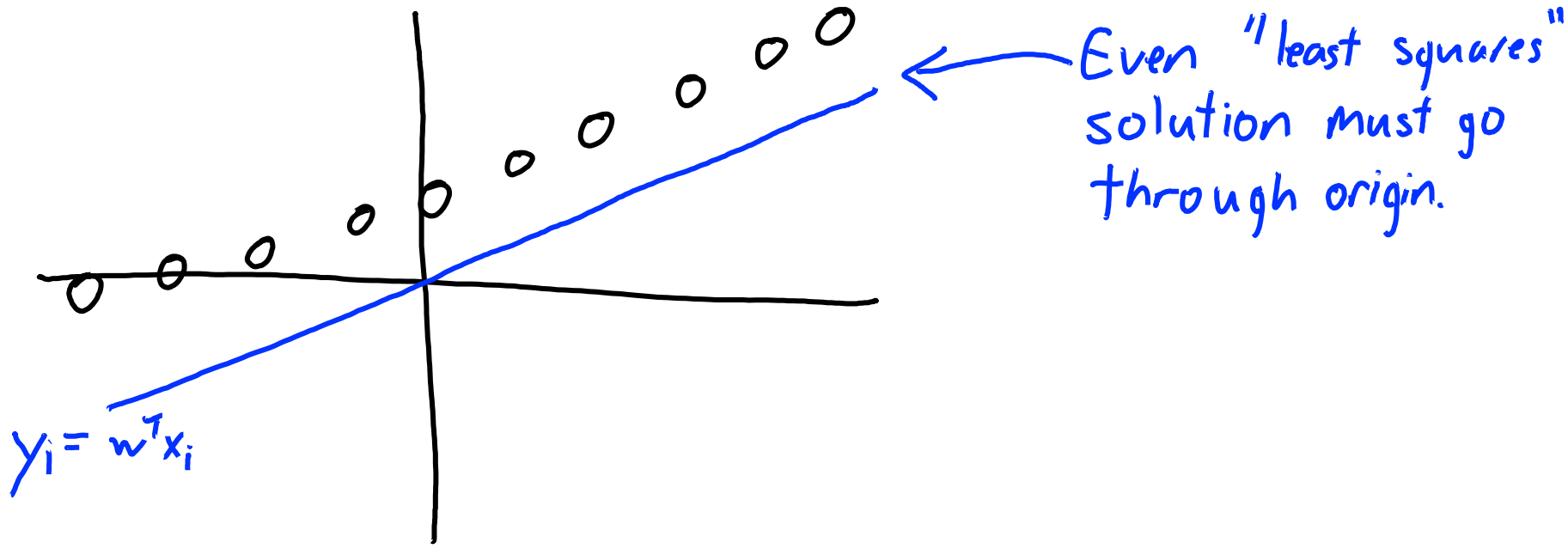
$$w = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_d \end{bmatrix}_{d \times 1} \qquad y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}_{n \times 1} \qquad x_i = \begin{bmatrix} x_{i1} \\ x_{i2} \\ \vdots \\ x_{id} \end{bmatrix}_{d \times 1}$$

- So feature matrix 'X' actually has $x_i$ transposed as rows:

$$X = \begin{bmatrix} — & x_1^T & — \\ — & x_2^T & — \\ & \vdots & \\ — & x_n^T & — \end{bmatrix}$$

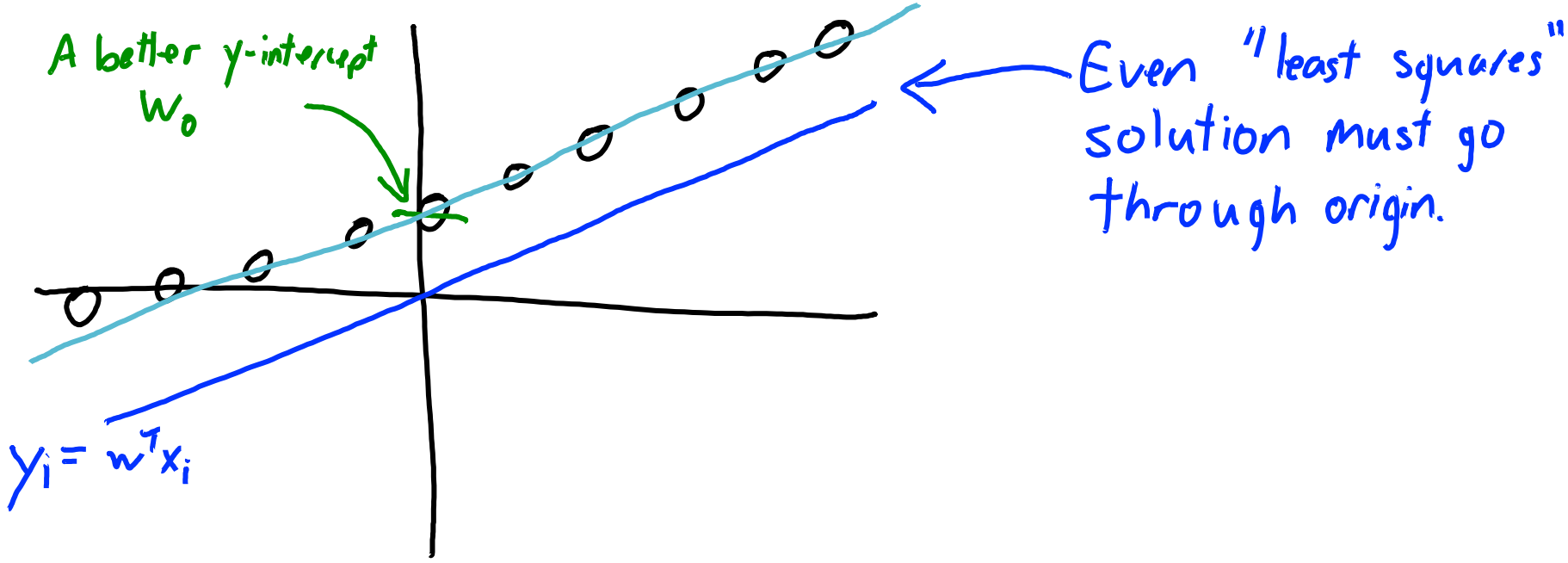
# Why don't we have a y-intercept?

- Last time: Linear models with no y-intercept.
  - Linear model is $y_i = w^T x_i$ instead of $y_i = w^T x_i + w_0$ with y-intercept $w_0$.
  - So if $x_i = 0$ then we must predict $y_i = 0$.



# Why don't we have a y-intercept?

- Last time: Linear models with no y-intercept.
  - Linear model is $y_i = w^T x_i$ instead of $y_i = w^T x_i + w_0$ with y-intercept $w_0$.
  - So if $x_i = 0$ then we must predict $y_i = 0$.



# Adding a Bias Variable

- Simple trick to add a y-intercept ("bias") variable:
  - Make a new matrix "Z" with an extra feature that is always "1".

$$X = \begin{bmatrix} 0.1 & 0.3 \\ 0.5 & -0.6 \\ 0.2 & 0.4 \end{bmatrix} \qquad Z = \begin{bmatrix} 1 & 0.1 & 0.3 \\ 1 & 0.5 & -0.6 \\ 1 & 0.2 & 0.4 \end{bmatrix}$$

- Now use "Z" as features to get a model with a non-zero y-intercept:

$$y_i = w_0 z_{i0} + w_1 z_{i1} + w_2 z_{i2}$$

($z_{i0}$ → "1", $z_{i1}$ → $x_{i1}$, $z_{i2}$ → $x_{i2}$)

$$= w_0 + w_1 x_{i1} + w_2 x_{i2}$$

- So we can have a non-zero y-intercept by changing features.

# Linear Least Squares

Prediction:

$$y = X * w$$

Why?

$y_i = w^T x_i$ so

$$y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} w^T x_1 \\ w^T x_2 \\ \vdots \\ w^T x_n \end{bmatrix} = \begin{bmatrix} x_1^T w \\ x_2^T w \\ \vdots \\ x_n^T w \end{bmatrix} = \underbrace{\begin{bmatrix} — & x_1^T & — \\ — & x_2^T & — \\ & \vdots & \\ — & x_n^T & — \end{bmatrix}}_{X} \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_d \end{bmatrix} = Xw$$

# Linear Least Squares

- We can rewrite the objective function as a norm

Want '$w$' that minimizes

$$f(w) = \frac{1}{2}\sum_{i=1}^{n}(w^T x_i - y_i)^2 = \frac{1}{2}\sum_{i=1}^{n} r_i^2 = \frac{1}{2} r^T r = \frac{1}{2}\|r\|_2^2 = \boxed{\frac{1}{2}\|Xw - y\|^2}$$

where $r_i = w^T x_i - y_i$

Define "residual" $r_i$ as signed error on example '$i$':

$$r_i = w^T x_i - y_i$$

$$r = \begin{bmatrix} r_1 \\ r_2 \\ \vdots \\ r_n \end{bmatrix} = \begin{bmatrix} w^T x_1 - y_1 \\ w^T x_2 - y_2 \\ \vdots \\ w^T x_n - y_n \end{bmatrix} = \underbrace{\begin{bmatrix} w^T x_1 \\ w^T x_2 \\ \vdots \\ w^T x_n \end{bmatrix}}_{Xw} - \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = Xw - y$$

# Linear Least Squares

Want 'w' that minimizes

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

Let's expand then compute gradient.

$$= \frac{1}{2}(Xw - y)^T(Xw - y)$$

$$= \frac{1}{2}((Xw)^T - y^T)(Xw - y)$$

$$= \frac{1}{2}(w^T X^T - y^T)(Xw - y)$$

$$= \frac{1}{2}(w^T X^T(Xw - y) - y^T(Xw - y))$$

$$= \frac{1}{2}(w^T X^T Xw - w^T X^T y - y^T Xw + y^T y)$$

$$= \frac{1}{2}w^T X^T Xw - w^T X^T y + \frac{1}{2}y^T y$$

A good way to check your steps: make sure that dimensions all make sense.

# Linear Least Squares

See notes on linear and quadratic derivatives for details.

Training: `w = solve(X.T @ X, x.T @ y)`

Why?

Want 'w' that minimizes

$$f(w) = \frac{1}{2}\sum_{i=1}^{n}(w^T x_i - y_i)^2 = \frac{1}{2}\|Xw - y\|_2^2 = \frac{1}{2}w^T X^T X w - w^T X^T y + \frac{1}{2}y^T y$$

} a quadratic function (in matrix notation)

What are the gradients of these terms?

$$\nabla f(w) = X^T X w - X^T y + 0$$

So at a minimizer where $\nabla f(w) = 0$ we have: $\boxed{X^T X w = X^T y}$ "normal equations"

Some matrix  Some vector

This is a linear system $Aw = b$ for some matrix 'A' and vector 'b'

Cheat sheet:

$$\nabla_w[c] = 0$$

$$\nabla_w[w^T b] = b$$

This is like saying $\frac{d}{dw}[\alpha w] = \alpha$

$$\nabla_w\left[\frac{1}{2}w^T A w\right] = Aw \quad \text{for symmetric } A.$$

# The Punch Line

$$\min_w \frac{1}{2}||Xw - y||_2^2$$

```
w = solve(X.T @ X,  X.T @ y)
```

Note that
f(w) is a "convex" function
so solving $\nabla f(w) = 0$ gives minimizer

# Incorrect Solutions to Least Squares Problem

The least squares objective is $f(w) = \frac{1}{2}\|Xw - y\|^2$

The minimizers of this objective are solutions to the linear system:

$$X^T X w = X^T y$$

The following are not the solutions to the least squares problem:

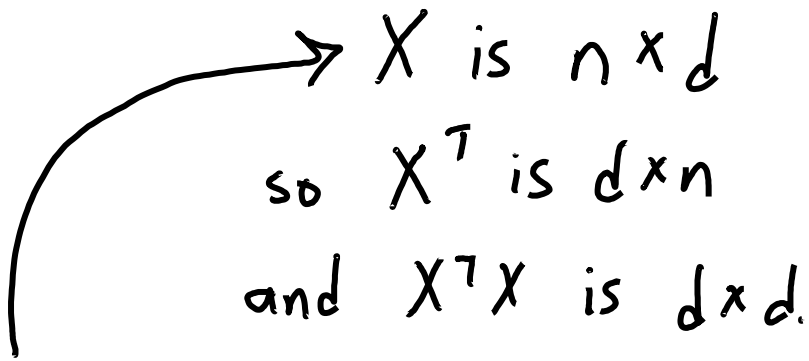
$w = (X^T X)^{-1}(X^T y)$ (only true if $X^T X$ is invertible)

$w X^T X = X^T y$ (matrix multiplication is not commutative, dimensions don't even match)

$w = \frac{X^T y}{X^T X}$ (you cannot divide by a matrix)

# Least Squares Issues

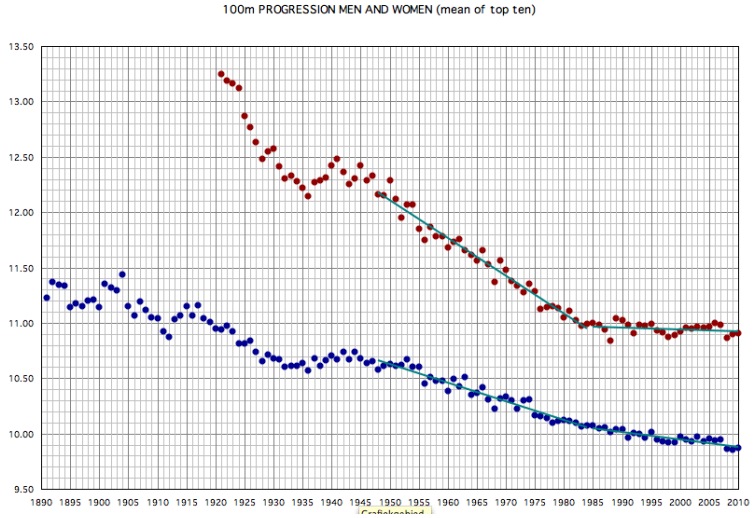
- Issues with least squares model:
  - Solution might not be unique.
  - It is sensitive to outliers.
  - It always uses all features.
  - Data can might so big we can't store $X^TX$.
  - It might predict outside range of $y_i$ values.
  - It assumes a linear relationship between $x_i$ and $y_i$.

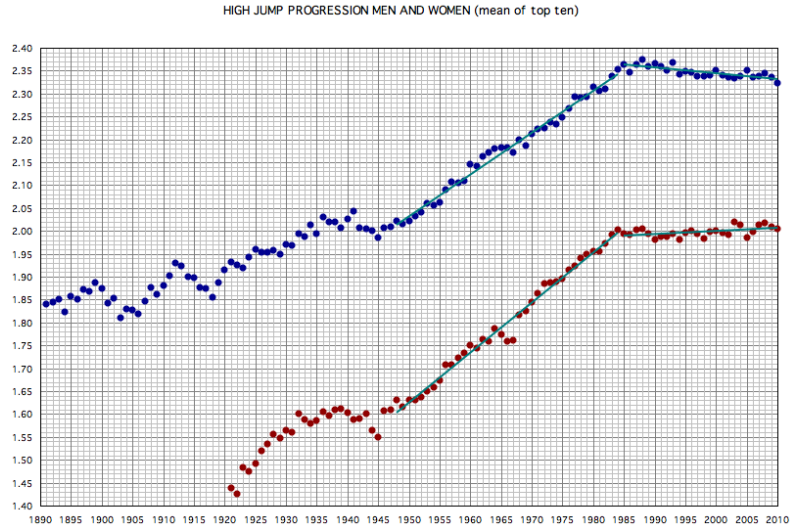


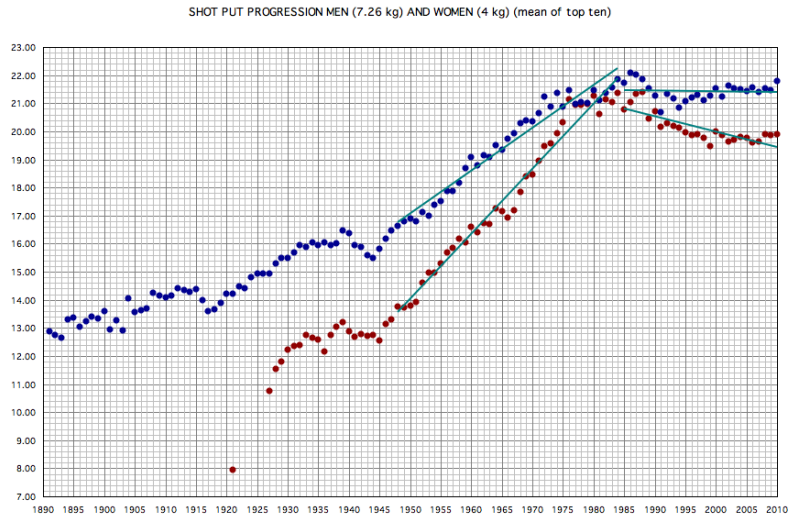

Costs $O(nd^2)$ to calculate:
- Each of the $O(d^2)$ elements is an inner product between length 'n' vectors.

# Example: Non-Linear Progressions in Athletics

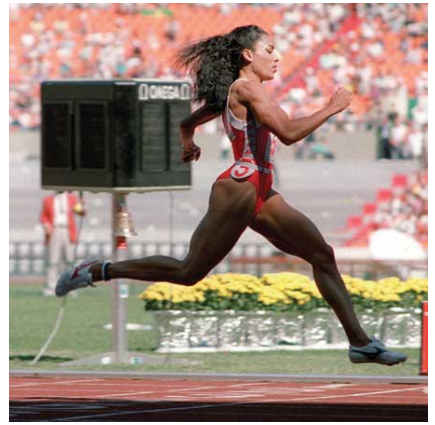
- Are top athletes going faster, higher, and farther?







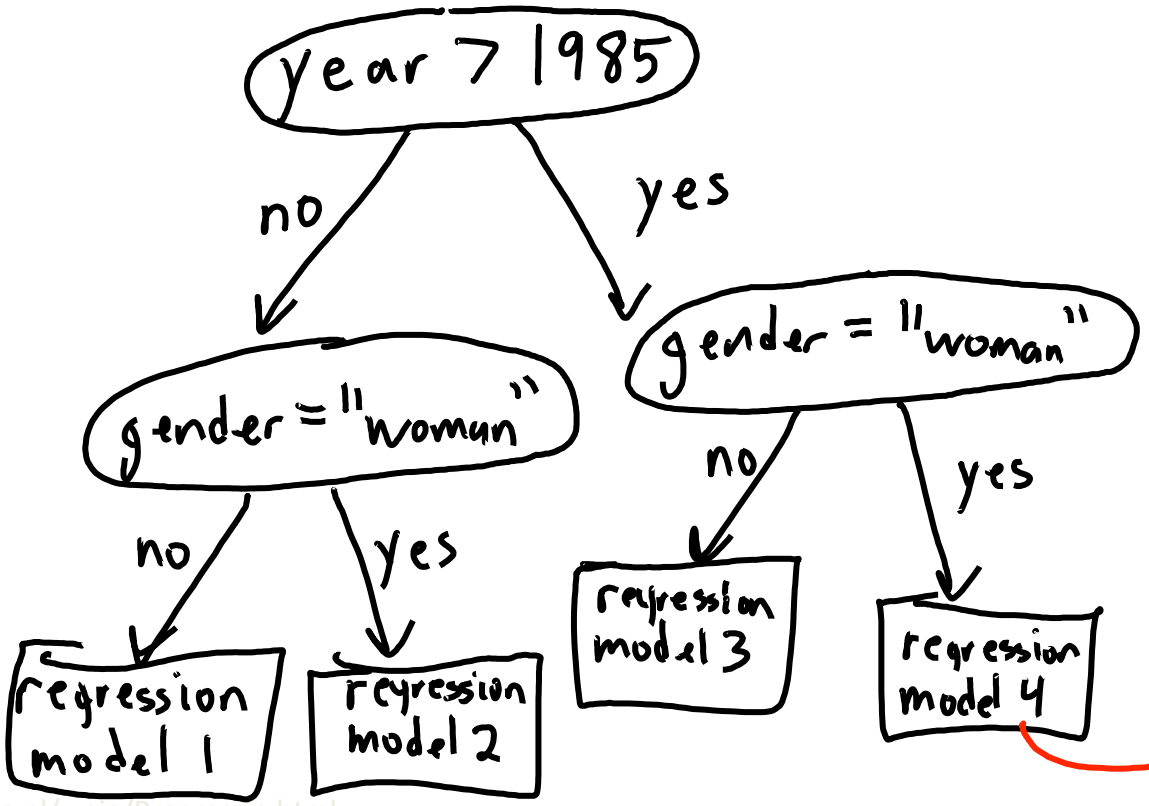
http://www.at-a-lanta.nl/weia/Progressie.html
https://en.wikipedia.org/wiki/Usain_Bolt
http://www.britannica.com/biography/Florence-Griffith-Joyner

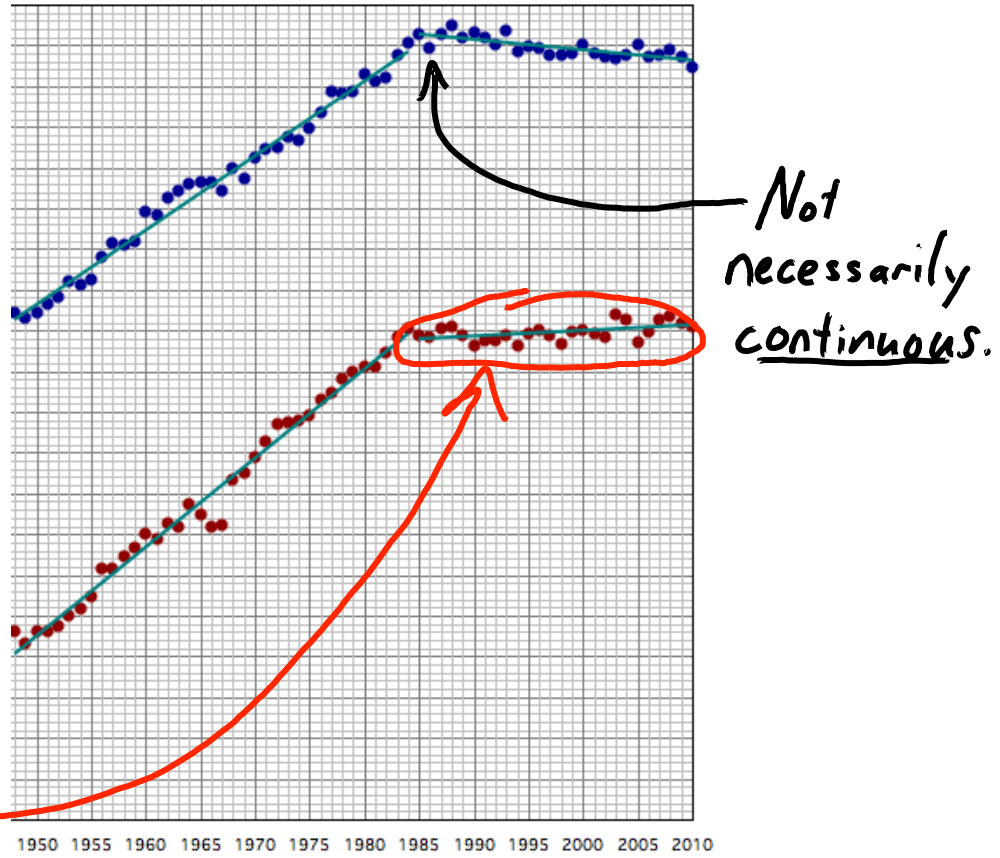
# Adapting Counting/Distance-Based Methods

- We can adapt our classification methods to perform regression:

# Adapting Counting/Distance-Based Methods

- We can adapt our classification methods to perform regression:
  - Regression tree: tree with mean value or linear regression at leaves.

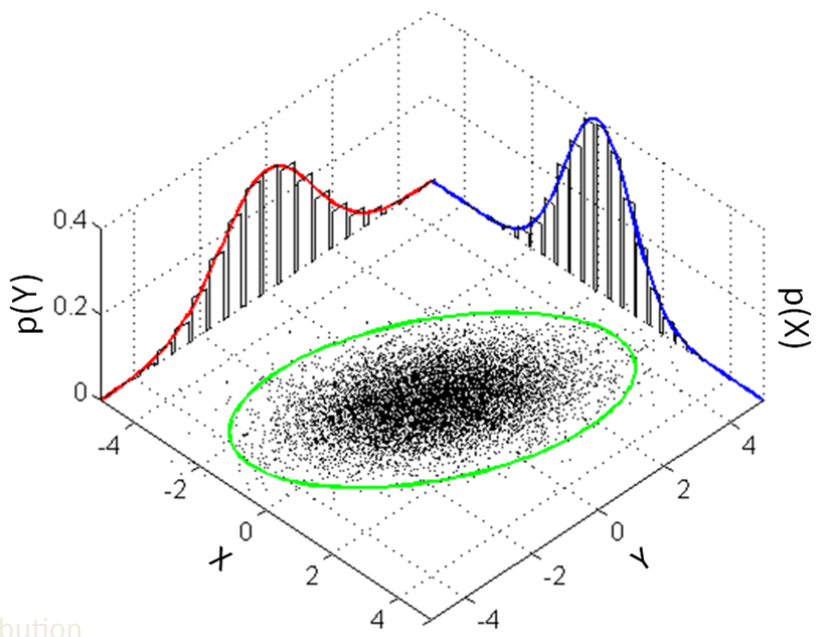



http://www.at-a-lanta.nl/weia/Progressie.html

# Adapting Counting/Distance-Based Methods

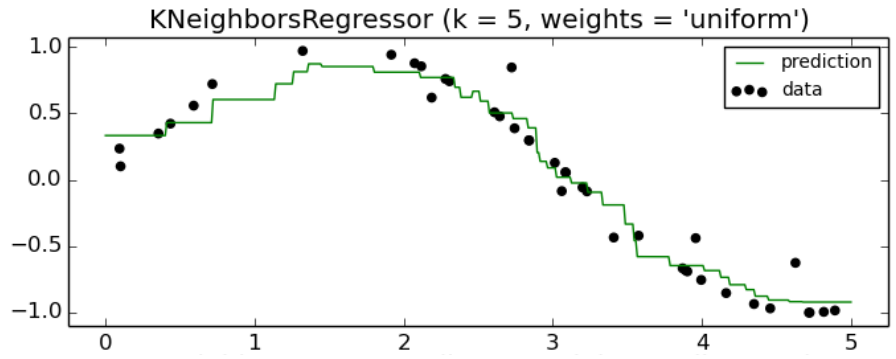
- We can adapt our classification methods to perform regression:
  - Regression tree: tree with mean value or linear regression at leaves.
  - Generative models: fit $p(x_i \mid y_i)$ and $p(y_i)$ with Gaussian or other model.




https://en.wikipedia.org/wiki/Multivariate_normal_distribution

# Adapting Counting/Distance-Based Methods

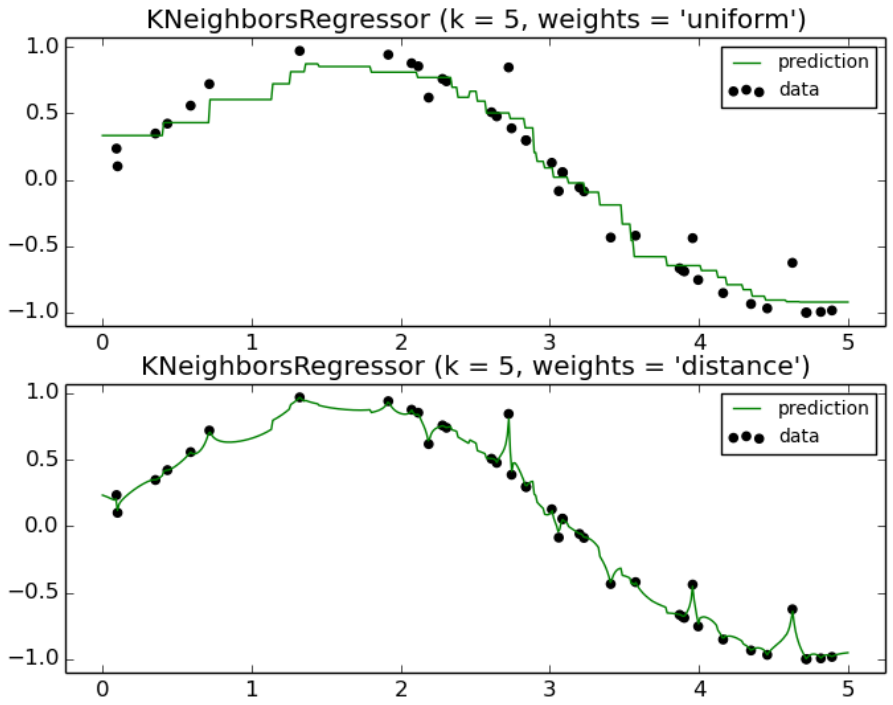
- We can adapt our classification methods to perform regression:
  - Regression tree: tree with mean value or linear regression at leaves.
  - Generative models: fit $p(x_i \mid y_i)$ and $p(y_i)$ with Gaussian or other model.
  - Non-parametric models:
    - Mean $y_i$ among k-nearest neighbours.




http://scikit-learn.org/stable/modules/neighbors.html

# Adapting Counting/Distance-Based Methods

- We can adapt our classification methods to perform regression:
  - Regression tree: tree with mean value or linear regression at leaves.
  - Generative models: fit $p(x_i \mid y_i)$ and $p(y_i)$ with Gaussian or other model.
  - Non-parametric models:
    - Mean $y_i$ among k-nearest neighbours.
    - Could be weighted by distance.
      - Close points 'j' get more "weight" $w_{ij}$.

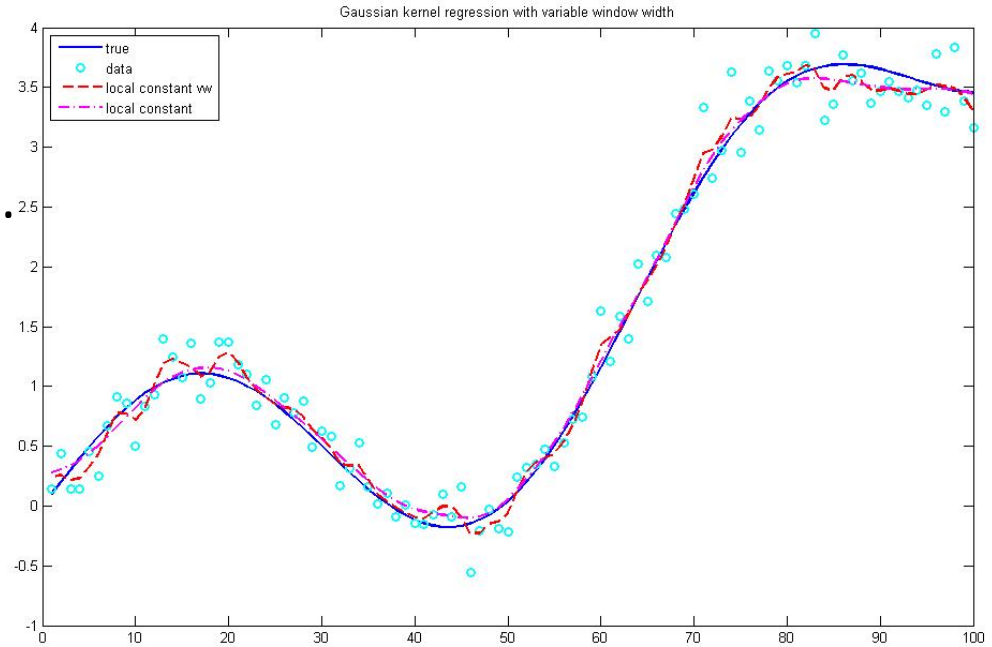



http://scikit-learn.org/stable/modules/neighbors.html

# Adapting Counting/Distance-Based Methods

- We can adapt our classification methods to perform regression:
  - Regression tree: tree with mean value or linear regression at leaves.
  - Generative models: fit $p(x_i \mid y_i)$ and $p(y_i)$ with Gaussian or other model.
  - Non-parametric models:
    - Mean $y_i$ among k-nearest neighbours.
    - Could be weighted by distance.
    - 'Nadaraya-Waston': weight *all* $y_i$ by distance to $x_i$.

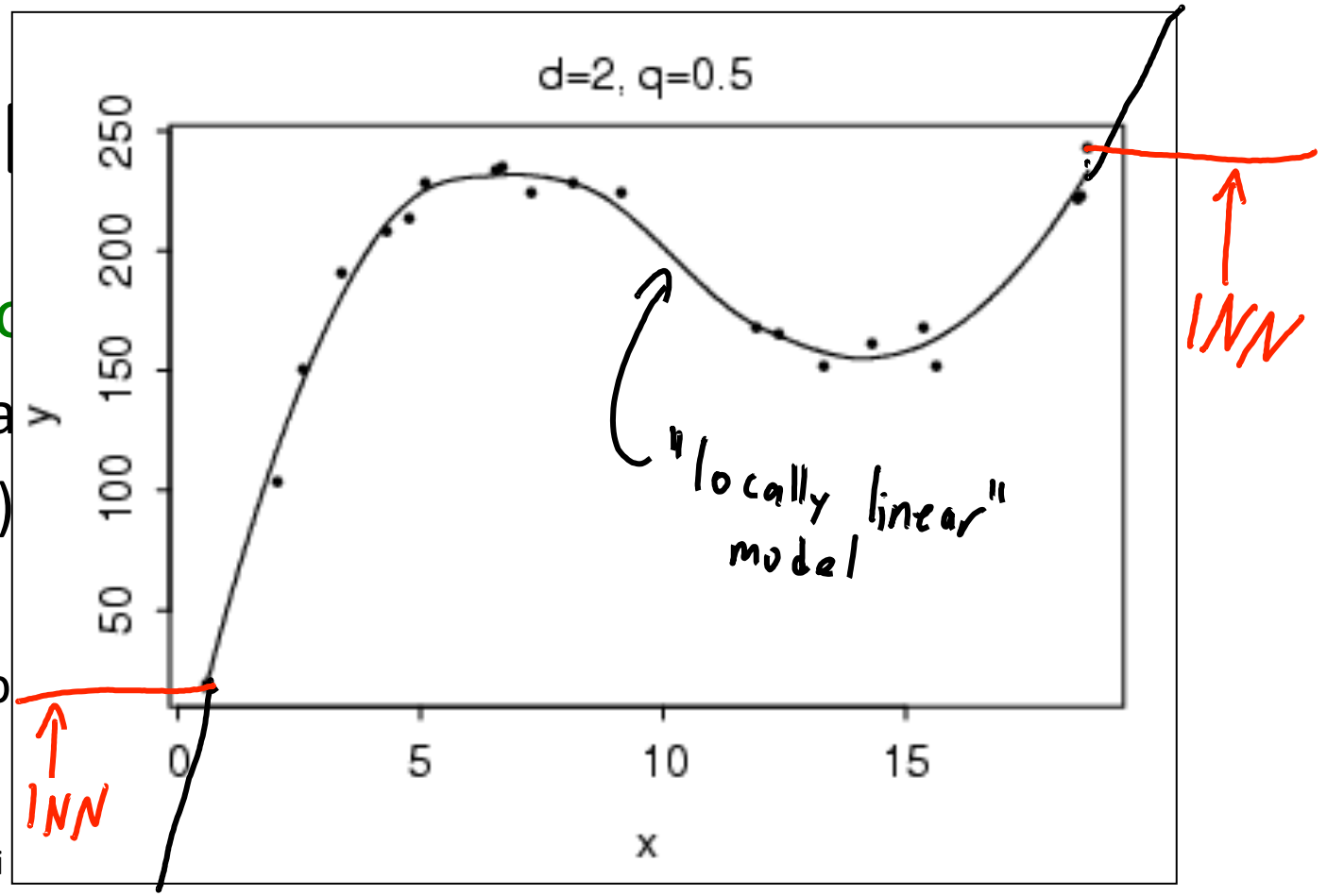
$$y_i = \frac{\sum_{j=1}^{n} w_{ij} y_j}{\sum_{j=1}^{n} w_{ij}}$$




http://www.mathworks.com/matlabcentral/fileexchange/35316-kernel-regression-with-variable-window-width/content/ksr_vw.m

# Adapting Counting/

- We can adapt our classificatio
  - Regression tree: tree with mea
  - Generative models: fit $p(x_i \mid y_i)$
  - Non-parametric models:
    - Mean $y_i$ among k-nearest neighb
    - Could be weighted by distance.
    - 'Nadaraya-Waston': weight *all* $y_i$
    - 'Locally linear regression': for each $x_i$ a fit linear model weighted by distance.
      (Better than KNN and NW at boundaries.)




http://www.itl.nist.gov/div898/handbook/pmd/section4/pmd423.htm

# Adapting Counting/Distance-Based Methods

- We can adapt our classification methods to perform regression:
  - Regression tree: tree with mean value or linear regression at leaves.
  - Generative models: fit $p(x_i \mid y_i)$ and $p(y_i)$ with Gaussian or other model.
  - Non-parametric models:
    - Mean $y_i$ among k-nearest neighbours.
    - Could be weighted by distance.
    - 'Nadaraya-Waston': weight *all* $y_i$ by distance to $x_i$.
    - 'Locally linear regression': for each $x_i$, fit linear model weighted by distance.
      (Better than KNN and NW at boundaries.)
  - Ensemble methods:
    - Can improve performance by averaging across regression models.

http://www.itl.nist.gov/div898/handbook/pmd/section4/pmd423.htm

# Linear Least Squares for Quadratic Models

- Can we use linear least squares to fit a quadratic model?

$$y_i = w_0 + w_1 x_i + w_2 x_i^2$$

- You can do this by changing the features (change of basis):

$$X = \begin{bmatrix} 0.2 \\ -0.5 \\ 1 \\ 4 \end{bmatrix} \qquad Z = \begin{bmatrix} 1 & 0.2 & (0.2)^2 \\ 1 & -0.5 & (-0.5)^2 \\ 1 & 1 & (1)^2 \\ 1 & 4 & (4)^2 \end{bmatrix}$$

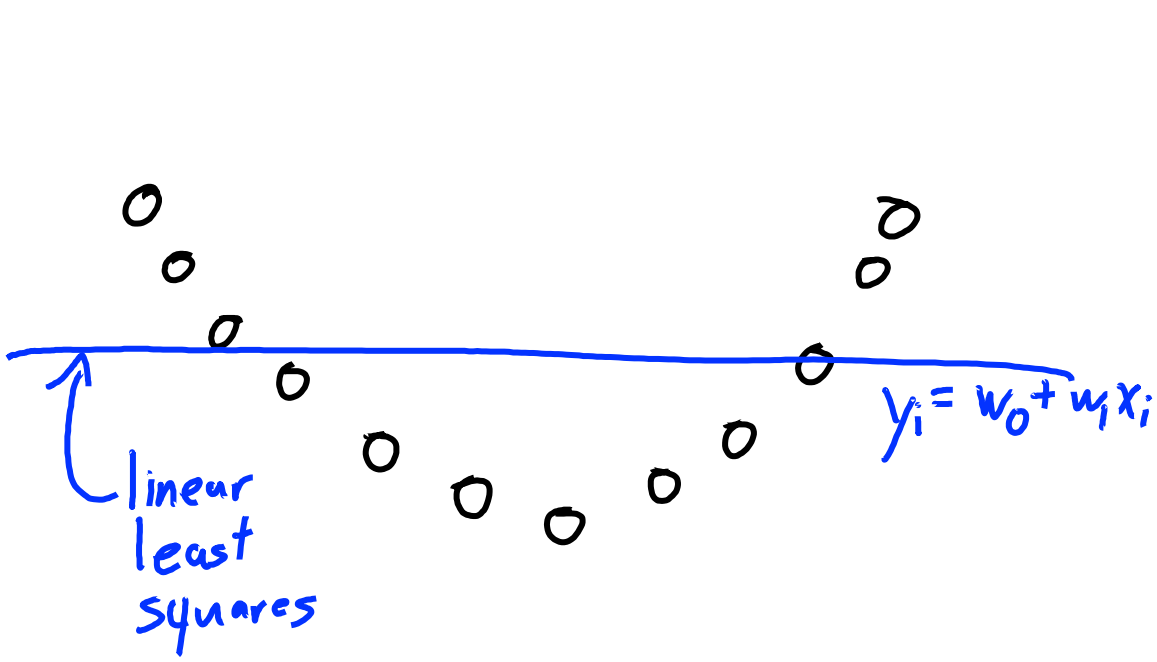
y-int x $x^2$

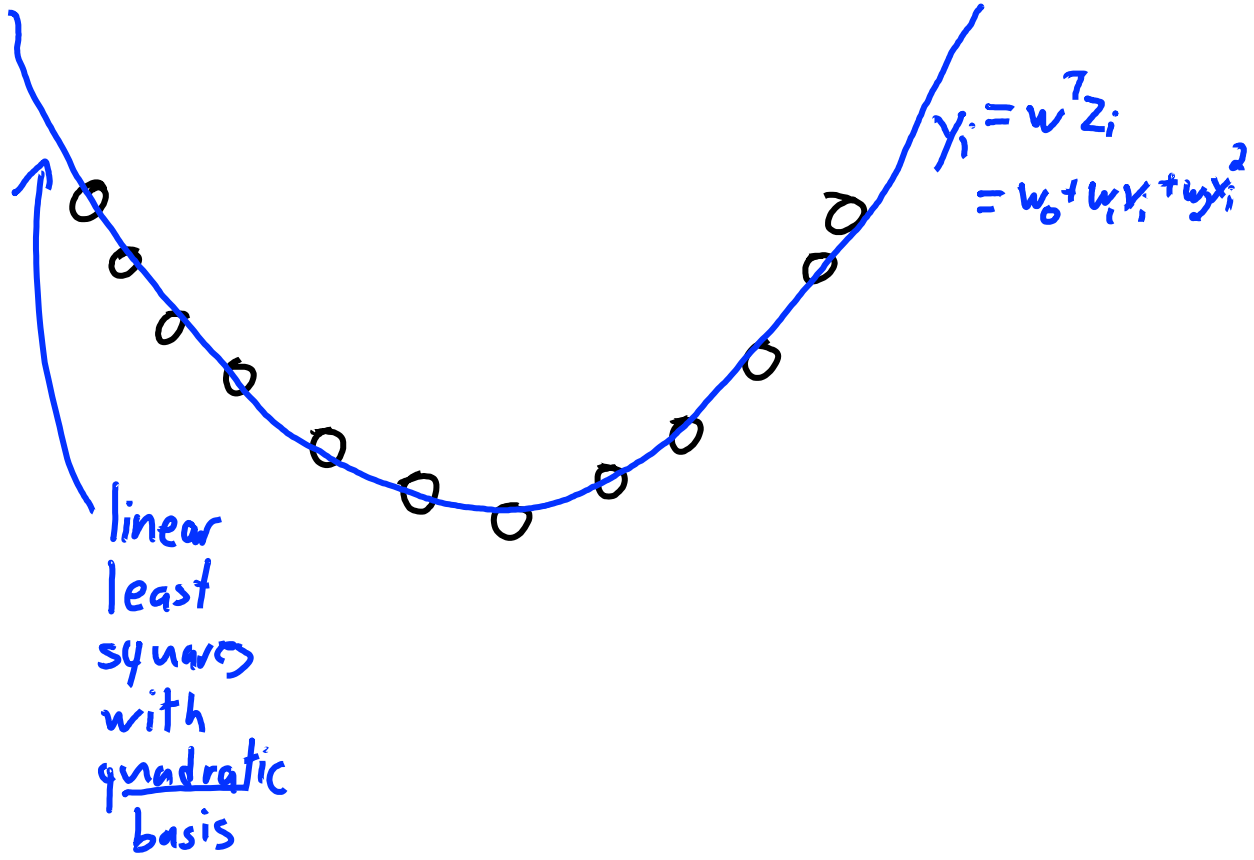
$$y_i = w^T z_i = w_0 z_{i0} + w_1 z_{i1} + w_2 z_{i2} = w_0 + w_1 x_i + w_2 x_i^2$$

- Fitting with least squares: $w = (Z^T Z) \backslash (Z^T y)$ ("solve linear system")
- It's a linear function of w, but a quadratic function of $x_i$.

To predict on new data $\hat{X}$, form $\hat{Z}$ from $\hat{X}$ and take $\hat{y} = \hat{Z}w$

# Linear Least Squares for Quadratic Models



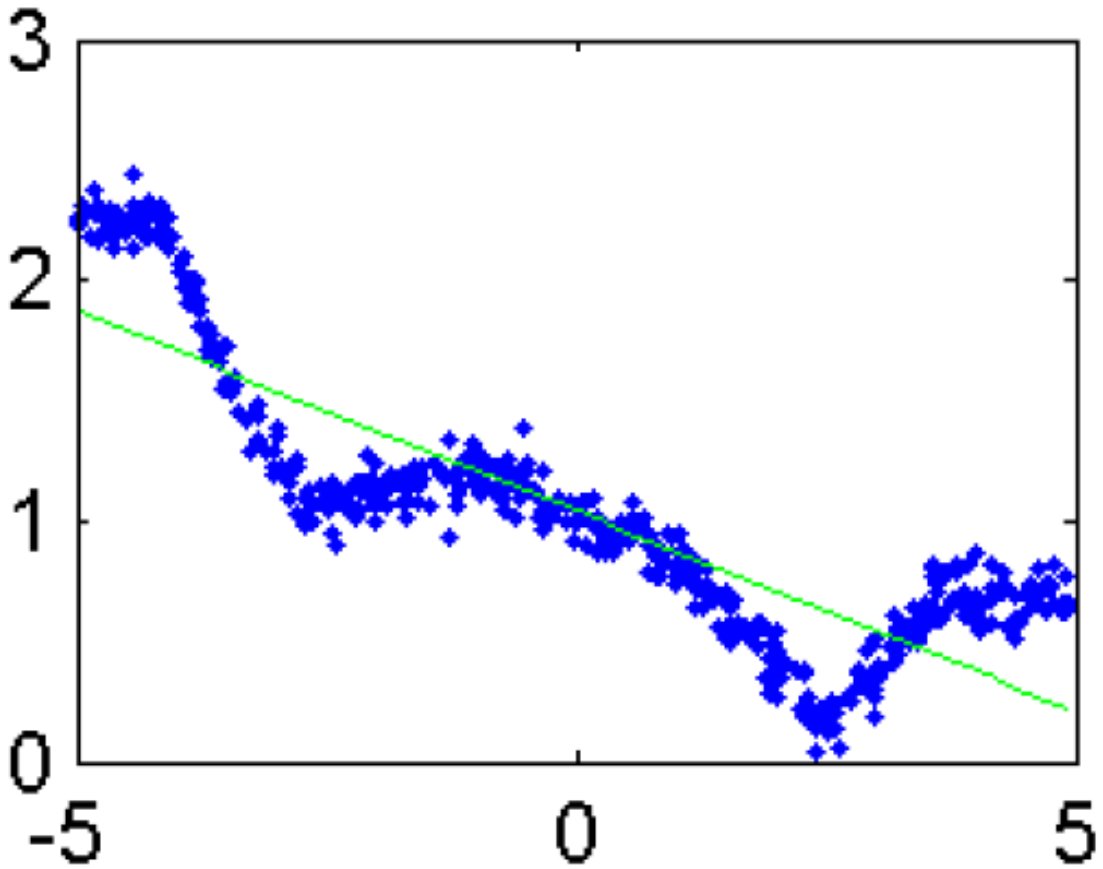
# General Polynomial Basis

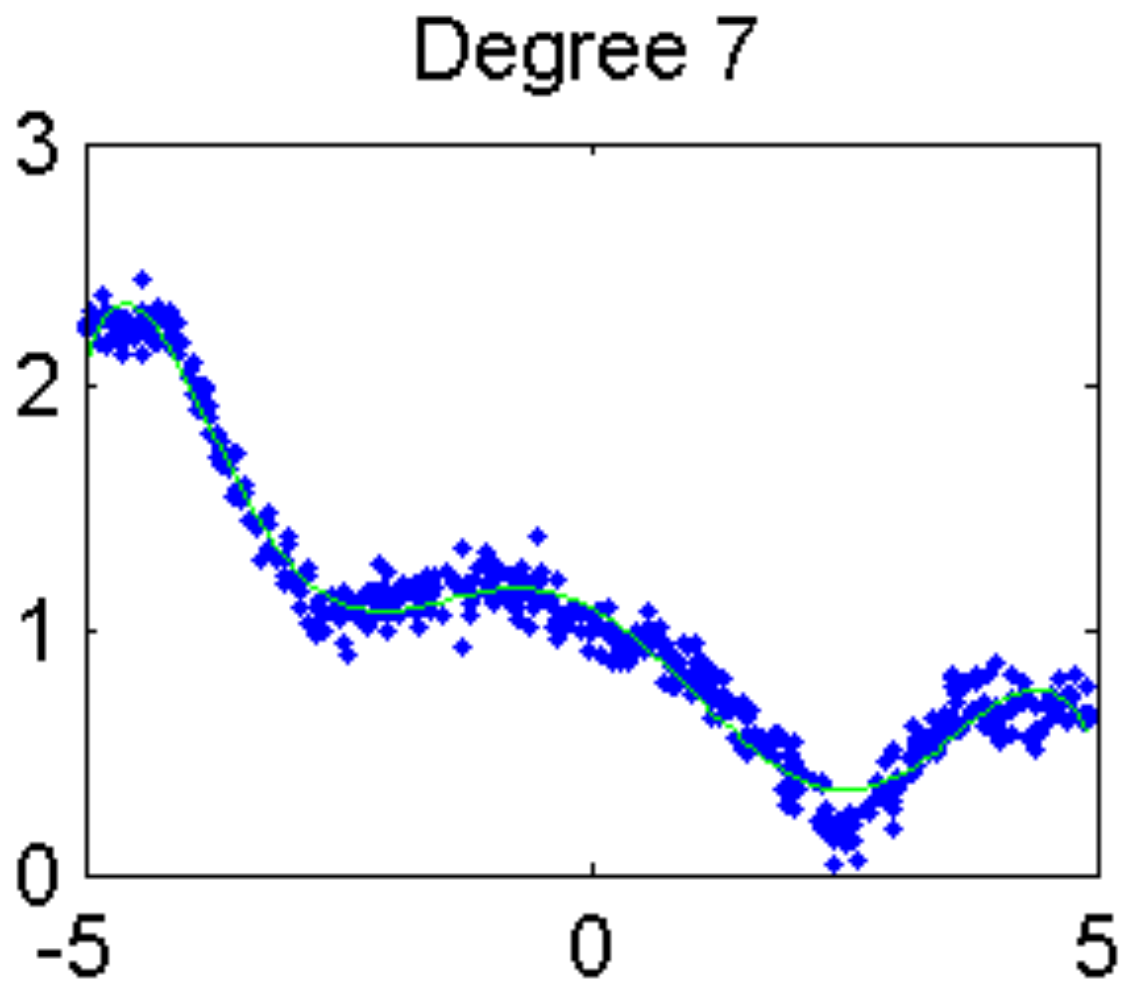
- We can have a polynomial of degree 'p' by using a basis:

$$Z = \begin{bmatrix} 1 & x_1 & (x_1)^2 & \cdots & (x_1)^p \\ 1 & x_2 & (x_2)^2 & \cdots & (x_2)^p \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & (x_n)^2 & \cdots & (x_n)^p \end{bmatrix}$$

- There are polynomial basis functions that are numerically nicer:
  - E.g., Lagrange polynomials (see CPSC 303)

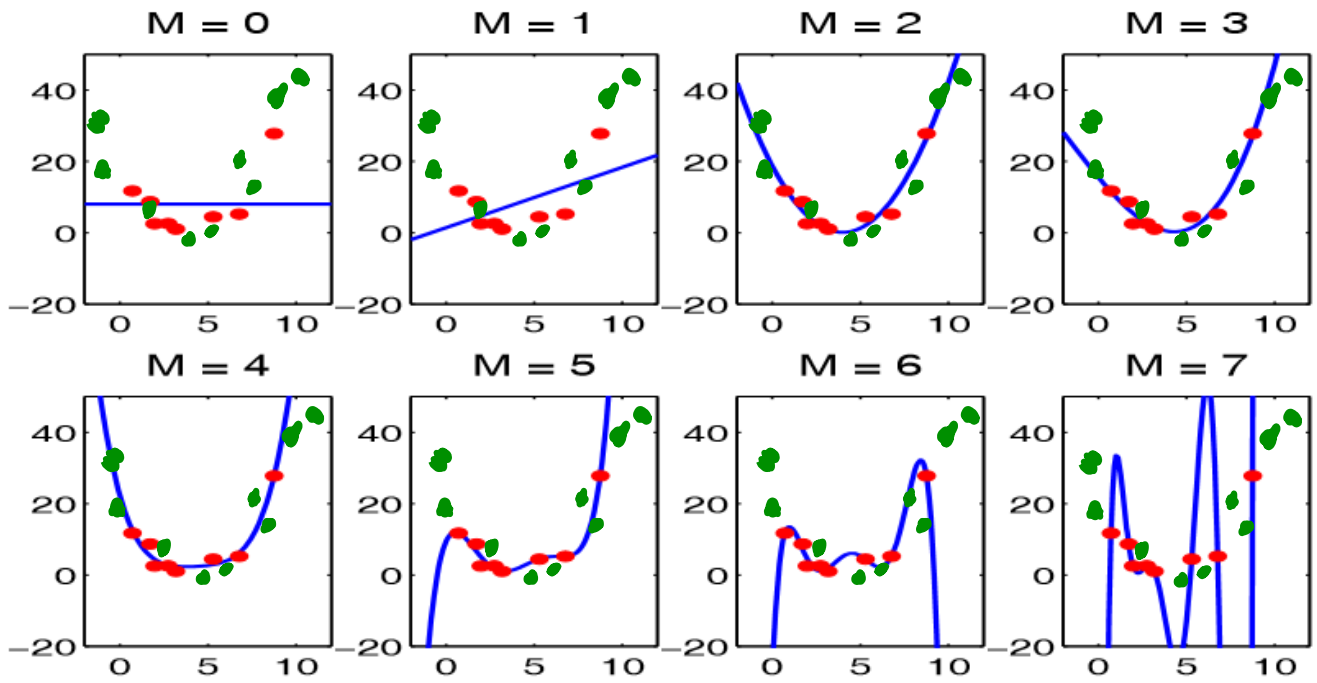
# General Polynomial Basis



# Degree of Polynomial and Fundamental Trade-Off

- As the polynomial degree increases, the training error goes down.




- But training error becomes worse approximation test error.
- Usual approach to selecting degree: validation or cross-validation.

# Summary

- Y-intercept can be modeled by using a column of 1s.
- Linear least squares solution is given by normal equations:
  - Solve $(X^T X)w = X^T y$.
- Tree/generative/non-parametric/ensemble methods for regression.
- Change of basis allows linear models to model non-linear data

- Next time:
  - A general method for avoiding overfitting.