COMS 4771
Regression

Nakul Verma

Last time…

- Support Vector Machines
- Maximum Margin formulation
- Constrained Optimization
- Lagrange Duality Theory
- Convex Optimization
- SVM dual and Interpretation
- How get the optimal solution

Learning more Sophisticated Outputs

So far we have focused on classification $f : X \rightarrow \{1, \ldots, k\}$

What about **other outputs**?

- $PM_{2.5}$ (pollutant) particulate matter exposure estimate:

  Input: # cars, temperature, etc. **Output:** 50 ppb

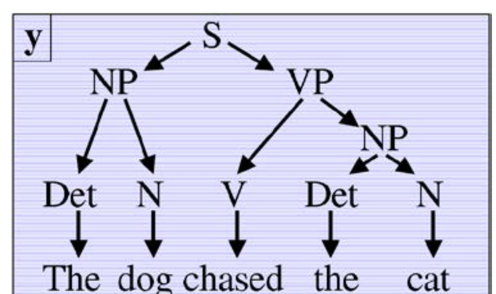
- Pose estimation

- Sentence structure estimate:

  x The dog chased the cat

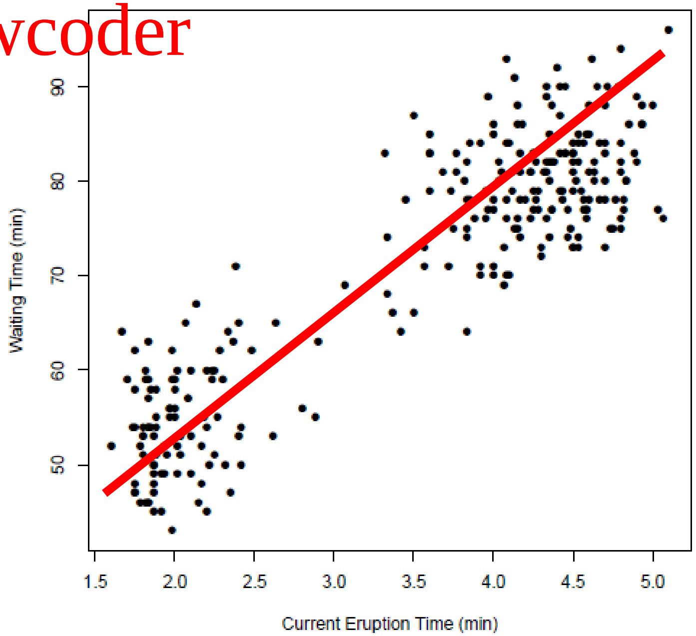
Regression

We'll focus on problems with real number outputs (regression problem):

$$f : X \to \mathbf{R}$$

Example:

Next eruption time of old faithful geyser (at Yellowstone)



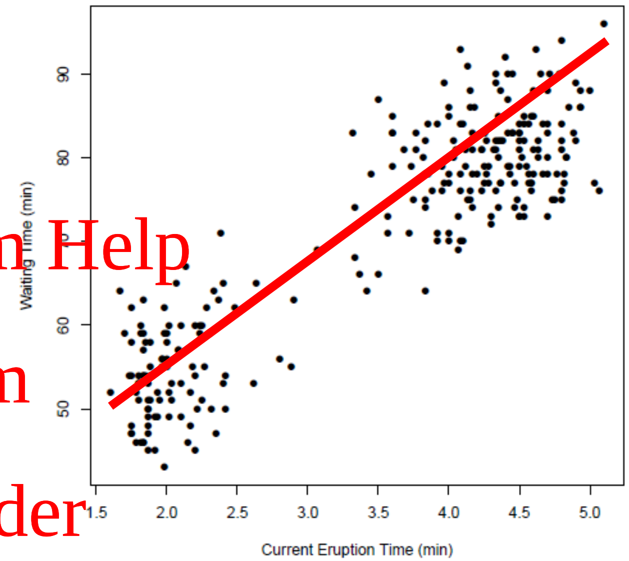
Regression Formulation for the Example

Given *x*, want to predict an estimate $\hat{y}$ of *y*, which minizes the discrepancy (*L*) between $\hat{y}$ and *y*.

$$L(\hat{y}; y) := |\hat{y} - y|$$ *Absolute error*

$$:= (\hat{y} - y)^2$$ *Squared error*

Loss



A **linear predictor** *f*, can be defined by the slope *w* and the intercept w_0 :

$$\hat{f}(\vec{x}) := \vec{w} \cdot \vec{x} + w_0$$

which minimizes the prediction loss.

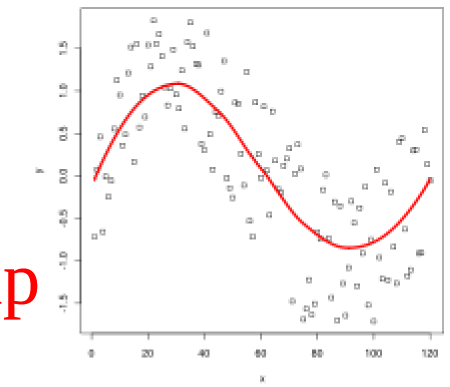
$$\min_{w, w_0} \mathbb{E}_{\vec{x}, y} \left[L(\hat{f}(\vec{x}), y) \right]$$

*How is this different from **classification**?*

Parametric vs non-parametric Regression

If we assume a particular form of the regressor:

Parametric regression

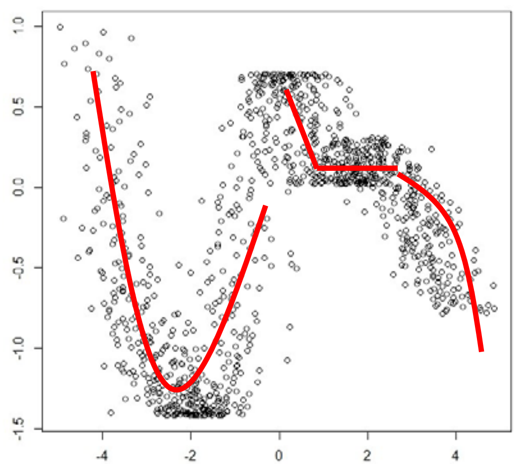


Goal: to learn the parameters which yield the minimum error/loss

If no specific form of regressor is assumed:

Non-parametric regression

Goal: to learn the predictor directly from the input data that yields the minimum error/loss



Linear Regression

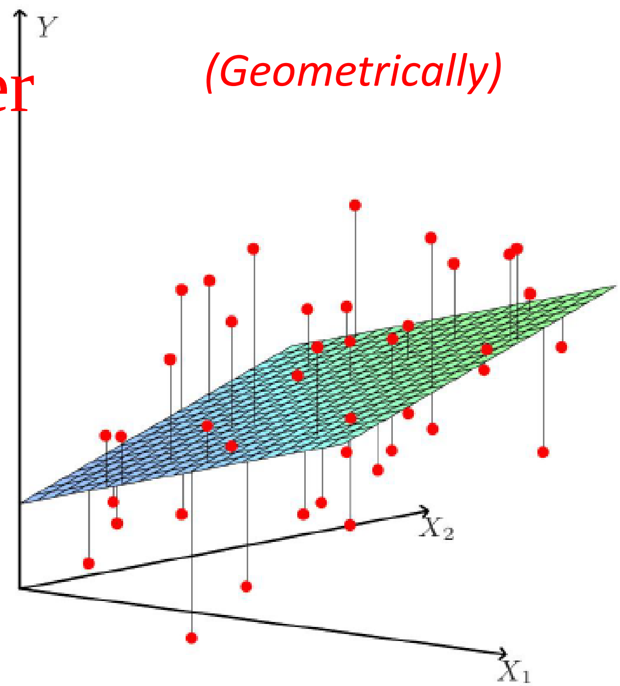
Want to find a **linear predictor** *f*, i.e., *w* (intercept w_0 absorbed via lifting):

$$\hat{f}(\vec{x}) := \vec{w} \cdot \vec{x}$$

which minimizes the prediction loss over the population.

$$\min_{\vec{w}} \mathbb{E}_{\vec{x},y}\left[L(\hat{f}(\vec{x}), y)\right]$$

(Geometrically)



We estimate the parameters by minimizing

the corresponding loss on the training data:

$$\arg\min_{w} \frac{1}{n} \sum_{i=1}^{n} \left[L(\vec{w} \cdot \vec{x}_i, y_i)\right]$$
$$= \arg\min_{w} \frac{1}{n} \sum_{i=1}^{n} \left(\vec{w} \cdot \vec{x}_i - y_i\right)^2$$

for squared error

Linear Regression: Learning the Parameters

Linear predictor with squared loss:

$$\arg\min_{w} \frac{1}{n} \sum_{i=1}^{n} \left(\vec{w} \cdot \vec{x}_i - y_i \right)^2$$

$$= \arg\min_{w} \left\| \begin{pmatrix} \ldots x_1 \ldots \\ \\ \ldots x_i \ldots \\ \\ \ldots x_n \ldots \end{pmatrix} \begin{bmatrix} \\ w \\ \\ \end{bmatrix} - \begin{bmatrix} y_1 \\ \\ y_i \\ \\ y_n \end{bmatrix} \right\|^2$$

$$= \arg\min_{w} \left\| X\vec{w} - \vec{y} \right\|_2^2$$

Unconstrained problem!

Can take the gradient and examine the stationary points!

Why need not check the second order conditions?

Linear Regression: Learning the Parameters

Best fitting *w*:

$$\frac{\partial}{\partial \vec{w}} \|X\vec{w} - \vec{y}\|^2 = 2X^\top(X\vec{w} - \vec{y})$$

$$X^\top X \vec{w} = X^\top \vec{y}$$

At a stationary point

$$\implies \vec{w}_{\mathrm{ols}} = (X^\top X)^\dagger X^\top \vec{y}$$

Pseudo-inverse

Also called the Ordinary Least Squares (OLS)

The solution is unique and stable when X^TX is invertible

What is the interpretation of this solution?

Linear Regression: Geometric Viewpoint

Consider the ***column space*** view of data **X**:

$$\begin{pmatrix} \cdots\ x_1\ \cdots \\ \cdots\ x_i\ \cdots \\ \cdots\ x_n\ \cdots \end{pmatrix} \qquad \ddot{x}_1, \ldots, \ddot{x}_d \in \mathbf{R}^n$$

Find a *w*, such that the linear combination of ***minimizes***

$$\frac{1}{n}\left\| \vec{y} - \sum_{i=1}^{d} w_i \ddot{x}_i \right\|^2 =: \text{residual} \qquad \hat{y} = X\vec{w}_{\text{ols}} = X(X^{\top}X)^{\dagger}X^{\top}\,\vec{y}$$

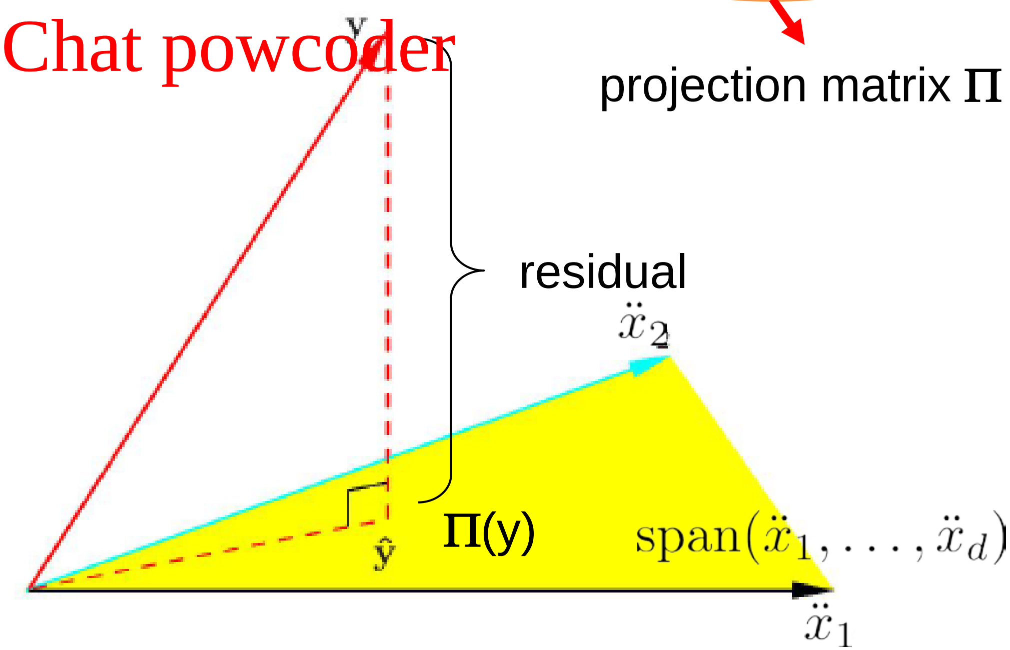
projection matrix Π

Say $\hat{y}$ is the ols solution, ie,

$$\hat{y} := X\vec{w}_{\text{ols}} = \sum_{i=1}^{d} w_{\text{ols},i}\ddot{x}_i$$

*Thus, $\hat{y}$ is the **orthogonal projection** of y onto the* $\text{span}(\ddot{x}_1, \ldots, \ddot{x}_d)$ *!*

*w_{ols} forms the **coefficients** of $\hat{y}$*



Linear Regression: Statistical Modeling View

Let’s assume that data is **generated** from the following process:

- A example x_i is draw independently from the data space **X**

$$x_i \sim \mathcal{D}_X$$

- y_{clean} is computed as $(w \cdot x_i)$, from a fixed unknown w

$$y_{\text{clean}} := w \cdot x_i$$

- y_{clean} is corrupted from by adding independent Gaussian noise $N(0,\sigma^2)$

$$y_i := y_{\text{clean}} + \epsilon_i = w \cdot x_i + \epsilon_i \qquad \epsilon_i \sim N(0, \sigma^2)$$

- (x_i, y_i) is revealed as the i^{th} sample

$$(x_1, y_1), \ldots, (x_n, y_n) =: S$$

Linear Regression: Statistical Modeling View

How can we determine *w*, from Gaussian noise corrupted observations?

$$S = (x_1, y_1), \ldots, (x_n, y_n)$$

Observation:

$$y_i \sim w \cdot x_i + N(0, \sigma^2) = N(w \cdot x_i, \sigma^2)$$

How to estimate parameters of a Gaussian?

parameter

Let's try Maximum Likelihood Estimation!

$$\log \mathcal{L}(w|S) = \sum_{i=1}^{n} \log p(y_i|w)$$

$$\propto \sum_{i=1}^{n} \frac{-(w \cdot x_i - y_i)^2}{2\sigma^2}$$

ignoring terms independent of w

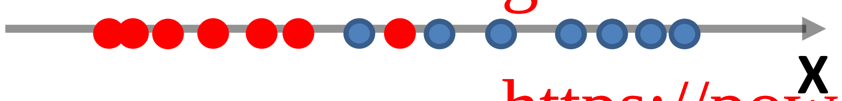
optimizing for w yields the same ols result!

What happens if we model each y_i with indep. noise of different variance?

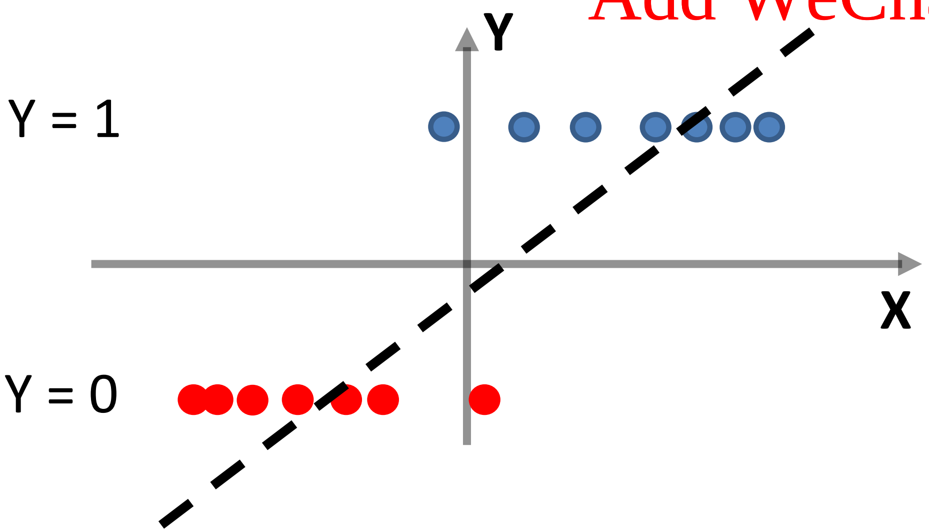
Linear Regression for Classification?

Linear regression seems general, can we use it to derive a binary classifier?

Let’s study 1-d data:



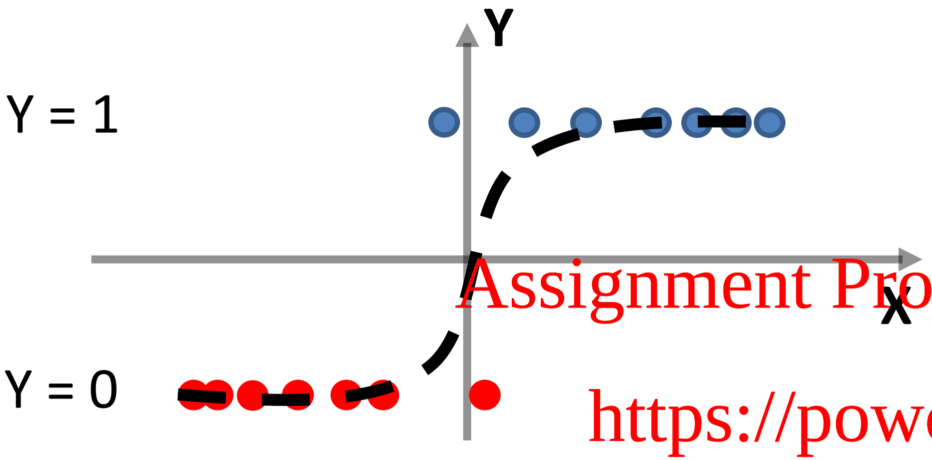
Problem #1: Where is y? for regression.



Problem #2: Not really linear!

*Perhaps it is linear in some **transformed** coordinates?*

Linear Regression for Classification



Sigmoid a better model!

$$\hat{y} = f(x) := \frac{1}{1 + e^{-w \cdot x}}$$

Binary predictor: $\mathrm{sign}(2f(x) - 1)$

Interpretation:

For an event that occurs with probability P, the ***odds*** of that event is:

$$\mathrm{odds}(P) := \frac{P}{1 - P}$$

For an event with P=0.9, odds = 9
But, for an event P=0.1, odds = 0.11
(very asymmetric)

Consider the "log" of the odds

$$\log(\mathrm{odds}(P)) := \mathrm{logit}(P) := \log\left(\frac{P}{1 - P}\right)$$

$$\mathrm{logit}(P) = -\mathrm{logit}(1 - P)$$

Symmetric!

Logistic Regression

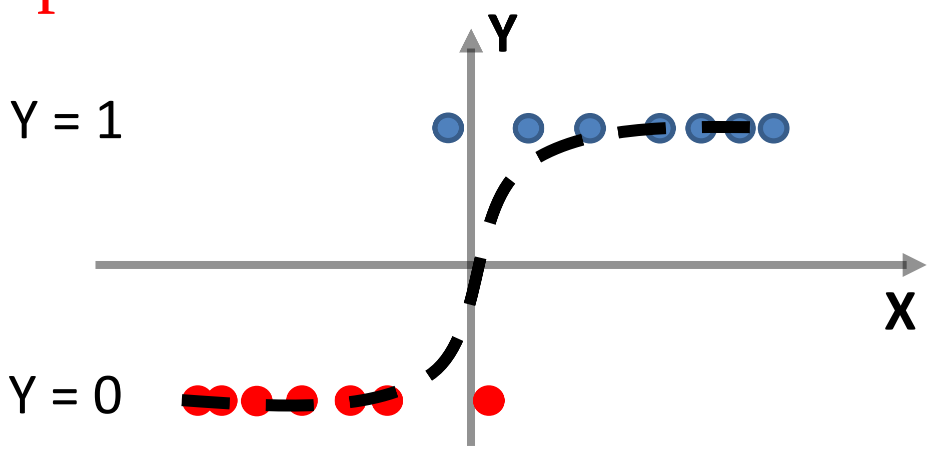
Model the log-odds or logit with linear function!

$$\text{logit}(P(x)) = \log\left(\frac{P(x)}{1-P(x)}\right) = w \cdot x$$

$$\frac{P(x)}{1-P(x)} = e^{w \cdot x}$$

$$P(x) = \frac{e^{w \cdot x}}{1+e^{w \cdot x}} = \frac{1}{1+e^{-w \cdot x}}$$

Sigmoid!



OK, we have a model, how do we learn the parameters?

Logistic Regression: Learning Parameters

Given samples $S = (x_1, y_1), \ldots, (x_n, y_n)$ ($y_i \in \{0,1\}$ binary)

$$\mathcal{L}(w|S) = \prod_{i=1}^{n} p(x_i)^{y_i} (1 - p(x_i))^{1-y_i}$$

Binomial

$$\log \mathcal{L}(w|S) = \sum_{i=1}^{n} y_i \log p(x_i) + (1 - y_i) \log(1 - p(x_i))$$

$$= \sum_{i=1}^{n} \log 1 - p(x_i) + \sum_{i=1}^{n} y_i \log \frac{p(x_i)}{1 - p(x_i)}$$

Now, use logistic model!

$$= \sum_{i=1}^{n} - \log 1 + e^{w \cdot x_i} + \sum_{i=1}^{n} y_i w \cdot x_i$$

Can take the derivative and analyze stationary points,
unfortunately no closed form solution
(use iterative methods like gradient descent to find the solution)

Linear Regression: Other Variations

Back to the ordinary least squares (ols):

$$\text{minimize} \quad \left\|X\vec{w} - \vec{y}\right\|_2^2$$

$$\vec{w}_{\text{ols}} = (X^\top X)^{-1} X^\top \vec{y}$$

Often poorly behaved when $X^T X$ not invertible

Additionally how can we incorporate prior knowledge?

- perhaps want *w* to be sparse. *Lasso regression*
- perhaps want to simple *w*. *Ridge regression*

Ridge Regression

Objective

$$\text{minimize } \|X\vec{w} - \vec{y}\|^2 + \lambda \|\vec{w}\|^2$$

reconstruction error

'regularization' parameter

$$\vec{w}_{\text{ridge}} = (X^\top X + \lambda I)^{-1} X^\top \vec{y}$$

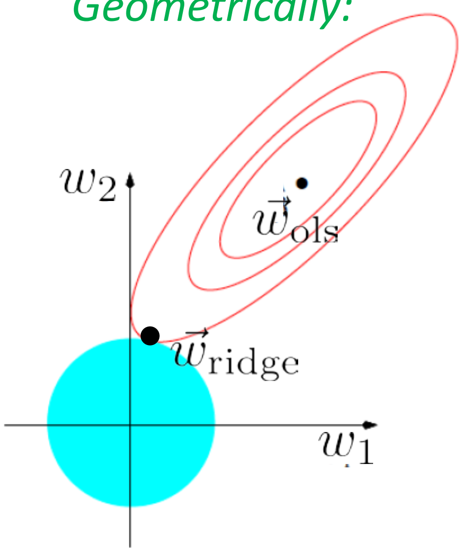
The 'regularization' helps avoid overfitting, and always resulting in a unique solution.

Geometrically:

Equivalent to the following optimization problem:

$$\begin{aligned} &\text{minimize } && \|X\vec{w} - \vec{y}\|^2 \\ &\text{such that } && \|\vec{w}\|^2 \leq B \end{aligned}$$

Why?



Lasso Regression

Objective

$$\text{minimize} \quad \left\|X\vec{w} - \vec{y}\right\|^2 + \lambda\|\vec{w}\|_1$$

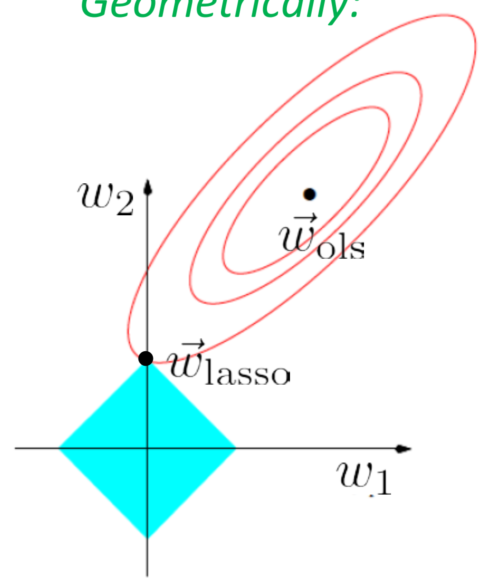
'lasso' penalty

$$\vec{w}_{\text{lasso}} = ?$$

no closed form solution

Lasso regularization encourages sparse solutions.

Geometrically:



Equivalent to the following optimization problem:

$$\begin{aligned} &\text{minimize} \quad \left\|X\vec{w} - \vec{y}\right\|^2 \\ &\text{such that} \quad \left\|\vec{w}\right\|_1 \leq B \end{aligned}$$

Why?

How can we find the solution?

What About Optimality?

Linear regression (and variants) is great, but what can we say about the best possible estimate?

Can we construct an estimator for real outputs that ***parallels*** *Bayes classifier for discrete outputs?*

Optimal L_2 Regressor

Best possible regression estimate at *x*: $f^*(x) := \mathbb{E}[Y|X=x]$

Theorem: for any regression estimate *g*(*x*)

$$\mathbb{E}_{(x,y)}|f^*(x) - y|^2 \leq \mathbb{E}_{(x,y)}|g(x) - y|^2$$

Similar to Bayes classifier, but for regression.

Proof is straightforward…

Proof

$$f^*(x) := \mathbb{E}[Y|X=x]$$

Consider L_2 error of $g(x)$

$$\mathbb{E}\big|g(x)-y\big|^2 = \mathbb{E}\big|g(x)-f^*(x)+f^*(x)-y\big|^2$$

$$= \mathbb{E}\big|g(x)-f^*(x)\big|^2 + \mathbb{E}\big|f^*(x)-y\big|^2$$

Why?

Cross term: $2\mathbb{E}\big[(g(x)-f^*(x))(f^*(x)-y)\big]$

$$= 2\mathbb{E}_x\Big[\mathbb{E}_{y|x}\big[(g(x)-f^*(x))(f^*(x)-y)\mid X=x\big]\Big]$$

$$= 2\mathbb{E}_x\Big[(g(x)-f^*(x))\,\mathbb{E}_{y|x}\big[(f^*(x)-y)\mid X=x\big]\Big]$$

$$= 2\mathbb{E}_x\big[(g(x)-f^*(x))(f^*(x)-f^*(x))\big] = 0$$

Therefore $\mathbb{E}\big|g(x)-y\big|^2 = \int_x \big|g(x)-f^*(x)\big|^2 \mu(dx) + \mathbb{E}\big|f^*(x)-y\big|^2$

Which is minimized when g(x) = f(x)!*

■

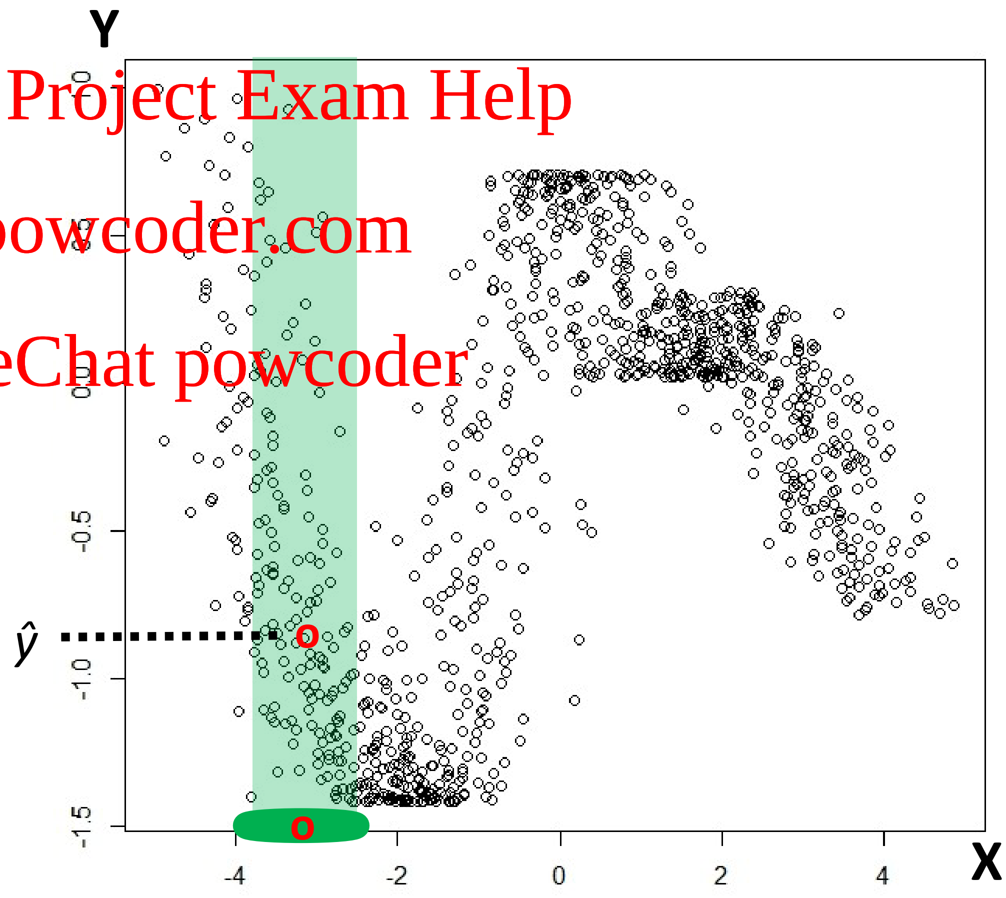
Non-parametric Regression

Linear regression (and variants) is great, but what if we don’t know parametric form of the relationship between the independent and dependent variables?

How can we predict value of a new test point *x* ***without*** model assumptions?

Idea:

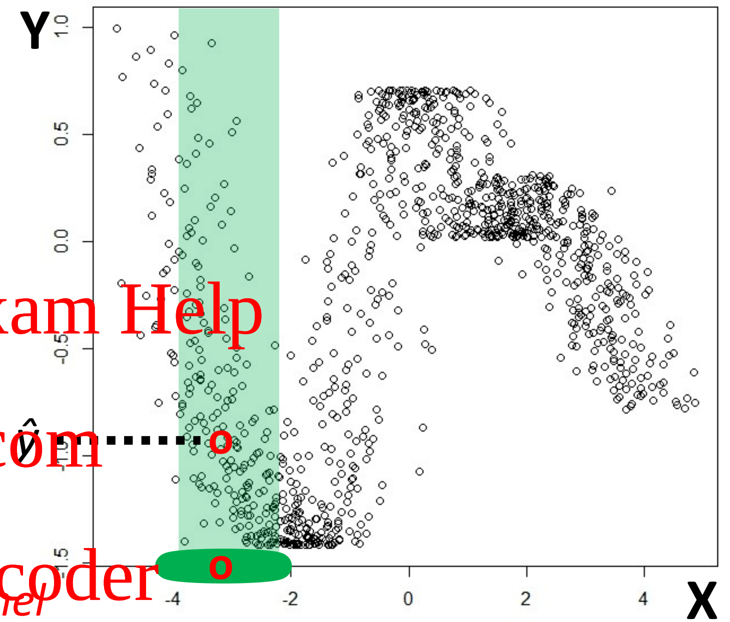
$\hat{y} = f(x) =$ *Average estimate* ***Y*** *of observed data in a local neighborhood* ***X*** *of x!*



Kernel Regression

$$\hat{y} = \hat{f}_n(x) := \sum_{i=1}^{n} w_i(x)\, y_i$$

Want weights that emphasize
local observations



Consider example localization functions:

$$K_h(x, x') = e^{-\|x - x'\|^2/h}$$ *Gaussian kernel*

$$= \mathbf{1}\left[\|x - x'\| \leq h\right]$$ *Box kernel*

$$= \left[1 - (1/h)\|x - x'\|\right]_+$$ *Triangle kernel*

Then define:

$$w_i(x) := \frac{K_h(x, x_i)}{\sum_{j=1}^{n} K_h(x, x_j)}$$ *Weighted average*

Consistency Theorem

Recall: best possible regression estimate at x: $f^*(x) := \mathbb{E}\big[Y|X=x\big]$

Theorem: As $n \to \infty$, $h \to 0$, $hn \to \infty$, then

$$\mathbb{E}_{(\vec{x},y)}\left|\hat{f}_{n,h}(x) - f^*(x)\right|^2 \to 0$$

where $\hat{f}_{n,h}(x) := \sum_{i=1}^{n} \frac{K_h(x,x_i)}{\sum_{j=1}^{n} K_h(x,x_j)} y_i$ is the kernel regressor with most localization kernels.

Proof is a bit tedious…

Proof Sketch

Prove for a fixed x and then integrate over (just like before)

$$\mathbb{E}\left|\hat{f}_{n,h}(x) - f^*(x)\right|^2 = \underbrace{\left[\mathbb{E}\hat{f}_{n,h}(x) - f^*(x)\right]^2}_{\text{squared bias of } \hat{f}_{n,h}} + \underbrace{\mathbb{E}\left[\hat{f}_{n,h}(x) - \mathbb{E}\hat{f}_{n,h}(x)\right]^2}_{\text{variance of } \hat{f}_{n,h}}$$

Bias-variance decomposition

Sq. bias $\approx c_1 h^2$

Variance $\approx c_2 \dfrac{1}{nh^d}$

Pick $h \approx n^{-1/2+d}$ $\qquad \mathbb{E}\left|\hat{f}_{n,h}(x) - f^*(x)\right|^2 \approx n^{-2/2+d} \to 0$

■

Kernel Regression

$$\hat{y} = \hat{f}_n(x) := \sum_{i=1}^{n} \frac{K_h(x, x_i)}{\sum_{j=1}^{n} K_h(x, x_j)} y_i$$

Advantages:

- Does not assume any parametric form of the regression function.
- Kernel regression is consistent

Disadvantages:

- Evaluation time complexity: O(*dn*)
- Need to keep all the data around!

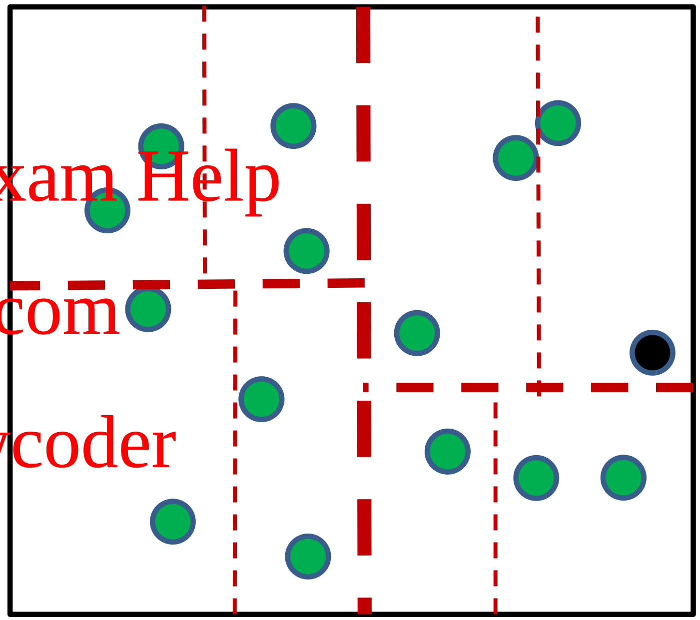
How can we address the shortcomings of kernel regression?

kd trees: Speed Up Nonparametric Regression

k-d trees to the rescue!

Idea: partition the data in cells organized in a tree based hierarchy. (just like before)

To return an estimated value, return the average y value in a cell!



What We Learned…

- Linear Regression
- Parametric vs Nonparametric regression
- Logistic Regression for classification
- Ridge and Lasso Regression
- Kernel Regression
- Consistency of Kernel Regression
- Speeding non-parametric regression with trees

Questions?

Next time…

Statistical Theory of Learning!