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
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

  $y$

  
  \[
  \begin{array}{cccc}
  \text{Det} & \text{N} & \text{V} & \text{Det} \\
  \text{The} & \text{dog} & \text{chased} & \text{the} \\
  \text{NP} & \text{VP} & \text{NP} \\
  \text{The} & \text{dog} & \text{chased} & \text{the} & \text{cat}
  \end{array}
  \]
We’ll focus on problems with real number outputs (regression problem):

\[ f : X \rightarrow \mathbb{R} \]

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 minimizes the discrepancy (\( L \)) between \( \hat{y} \) and \( y \).

Loss

\[
L(\hat{y}; y) := |\hat{y} - y| \quad \text{Absolute error} \\
:= (\hat{y} - y)^2 \quad \text{Squared error}
\]

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.

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

We estimate the parameters by minimizing the corresponding loss on the training data:

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

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

*for squared error*
Linear Regression: Learning the Parameters

Linear predictor with squared loss:

\[
\arg \min_w \frac{1}{n} \sum_{i=1}^{n} (\vec{w} \cdot \vec{x}_i - y_i)^2
\]

\[
= \arg \min_w \left\| \begin{pmatrix} \ldots x_1 \ldots \\
\ldots x_i \ldots \\
\ldots x_n \ldots 
\end{pmatrix} \begin{pmatrix} w \end{pmatrix} - \begin{pmatrix} y_1 \\
y_i \\
y_n \end{pmatrix} \right\|_2^2
\]

\[
= \arg \min_w \left\| X \vec{w} - \vec{y} \right\|_2^2
\]

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 $\mathbf{w}$:

$$\frac{\partial}{\partial \mathbf{w}} \| \mathbf{X} \mathbf{w} - \mathbf{y} \|^2 = 2 \mathbf{X}^\mathsf{T} (\mathbf{X} \mathbf{w} - \mathbf{y})$$

$$\mathbf{X}^\mathsf{T} \mathbf{X} \mathbf{w} = \mathbf{X}^\mathsf{T} \mathbf{y} \quad \text{At a stationary point}$$

$$\Rightarrow \mathbf{w}_{\text{ols}} = (\mathbf{X}^\mathsf{T} \mathbf{X})^\dagger \mathbf{X}^\mathsf{T} \mathbf{y} \quad \text{Also called the Ordinary Least Squares (OLS)}$$

Pseudo-inverse

The solution is unique and stable when $\mathbf{X}^\mathsf{T} \mathbf{X}$ is invertible

What is the interpretation of this solution?
Linear Regression: Geometric Viewpoint

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

$$\begin{bmatrix} \ldots \ x_1 \ldots \\ \ldots \ x_i \ldots \\ \ldots \ x_n \ldots \end{bmatrix} \quad \bar{x}_1, \ldots, \bar{x}_d \in \mathbb{R}^n$$

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

$$\frac{1}{n} \left\| \vec{y} - \sum_{i=1}^{d} w_i \bar{x}_i \right\|^2 =: \text{residual}$$

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

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

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

$w_{\text{ols}}$ forms the **coefficients** of $\hat{y}$
Let’s assume that data is generated from the following process:

- A example $x_i$ is draw independently from the data space $\mathbf{X}$
  \[ x_i \sim \mathcal{D}_X \]

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

- $y_{\text{clean}}$ is corrupted from by adding independent Gaussian noise $\mathcal{N}(0,\sigma^2)$
  \[ y_i := y_{\text{clean}} + \epsilon_i = w \cdot x_i + \epsilon_i \quad \epsilon_i \sim \mathcal{N}(0,\sigma^2) \]

- $(x_i, y_i)$ is revealed as the $i^{th}$ sample
  \[ (x_1, y_1), \ldots, (x_n, y_n) =: S \]
How can we determine $w$, from Gaussian noise corrupted observations?

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

Observation:

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

Let’s try Maximum Likelihood Estimation!

How to estimate parameters of a Gaussian?

Let’s try Maximum Likelihood Estimation!

parameter

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

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 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

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

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

Consider the “log” of the odds

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

$$\logit(P) = -\logit(1 - P)$$

Sigmoid a better model!

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

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

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

Symmetric!
Logistic Regression

Model the log-odds or logit with linear function:

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

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

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

\text{Sigmoid!}

\text{OK, we have a model, how do we learn the parameters?}
Logistic Regression: Learning Parameters

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

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

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

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

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

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

\[
\text{minimize } \|X\vec{w} - \vec{y}\|_2^2
\]

\[
\vec{w}_{\text{ols}} = (X^TX)^\dagger X^T\vec{y}
\]

Additionally how can we incorporate prior knowledge?

- perhaps want \( w \) to be sparse. \( \text{Lasso regression} \)
- perhaps want to simplify \( w \). \( \text{Ridge regression} \)

Often poorly behaved when \( X^TX \) not invertible
Ridge Regression

Objective

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

reconstruction error

'regularization' parameter

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

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

Equivalent to the following optimization problem:

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

such that $\| \vec{w} \|^2 \leq B$

Why?
Lasso Regression

Objective

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

‘lasso’ penalty

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

no closed form solution

Lasso regularization encourages sparse solutions.

Equivalent to the following optimization problem:

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

such that $\|\vec{w}\|_1 \leq B$

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 $g(x)$

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

Similar to Bayes classifier, but for regression.

*Proof is straightforward...*
Proof

Consider L₂ error of \( g(x) \)

\[
\mathbb{E} \left| g(x) - y \right|^2 = \mathbb{E} \left| g(x) - f^*(x) + f^*(x) - y \right|^2 \\
= \mathbb{E} \left| g(x) - f^*(x) \right|^2 + \mathbb{E} \left| f^*(x) - y \right|^2 \\
\]

**Cross term:**

\[
\begin{align*}
2\mathbb{E} \left[ (g(x) - f^*(x))(f^*(x) - y) \right] \\
&= 2\mathbb{E}_x \left[ \mathbb{E}_{y\mid x} \left[ (g(x) - f^*(x))(f^*(x) - y) \mid X = x \right] \right] \\
&= 2\mathbb{E}_x \left[ (g(x) - f^*(x)) \cdot \mathbb{E}_{y\mid x} \left[ (f^*(x) - y) \mid X = x \right] \right] \\
&= 2\mathbb{E}_x \left[ (g(x) - f^*(x))(f^*(x) - f^*(x)) \right] = 0
\end{align*}
\]

Therefore

\[
\mathbb{E} \left| g(x) - y \right|^2 = \int_x \left| g(x) - f^*(x) \right|^2 \mu(dx) + \mathbb{E} \left| f^*(x) - y \right|^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 \( x \) \textit{without} model assumptions?

Idea:

\[
\hat{y} = f(x) = \text{Average estimate } Y \text{ of observed data in a local neighborhood } X \text{ of } x!
\]
Kernel Regression

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

Want weights that emphasize \textbf{local} observations

Consider example localization functions:

\[
K_h(x, x') = e^{-\frac{\|x - x'\|^2}{2h}}
= 1 \left[ \|x - x'\| \leq h \right]
= \left[ 1 - \frac{1}{h} \|x - x'\| \right]_+
\]

Then define:

\[
w_i(x) := \frac{K_h(x, x_i)}{\sum_{j=1}^{n} K_h(x, x_j)} \quad \text{Weighted average}
\]
Recall: best possible regression estimate at $x$: $f^*(x) := \mathbb{E}[Y \mid X = x]$

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

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

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)

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

- squared bias of $\hat{f}_{n,h}$
- variance of $\hat{f}_{n,h}$

**Bias-variance decomposition**

Sq. bias $\approx c_1 h^2$

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

Pick $h \approx n^{-1/2+d}$

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