Outline

- Linear regression
- Ridge regression
- Logistic regression
- Other finite-sum models
Linear Regression
Regression

- Input: training data $x_1, x_2, \ldots, x_n \in \mathbb{R}^d$ and corresponding outputs $y_1, y_2, \ldots, y_n \in \mathbb{R}$
- Training: compute a function $f$ such that $f(x_i) \approx y_i$ for all $i$
- Prediction: given a testing sample $\tilde{x}$, predict the output as $f(\tilde{x})$

Examples:
- Income, number of children $\Rightarrow$ Consumer spending
- Processes, memory $\Rightarrow$ Power consumption
- Financial reports $\Rightarrow$ Risk
- Atmospheric conditions $\Rightarrow$ Precipitation
Regression

- Input: training data $x_1, x_2, \ldots, x_n \in \mathbb{R}^d$ and corresponding outputs $y_1, y_2, \ldots, y_n \in \mathbb{R}$
- Training: compute a function $f$ such that $f(x_i) \approx y_i$ for all $i$
- Prediction: given a testing sample $\tilde{x}$, predict the output as $f(\tilde{x})$
- Examples:
  - Income, number of children $\Rightarrow$ Consumer spending
  - Processes, memory $\Rightarrow$ Power consumption
  - Financial reports $\Rightarrow$ Risk
  - Atmospheric conditions $\Rightarrow$ Precipitation
Linear Regression

- Assume $f(\cdot)$ is a **linear function** parameterized by $w \in \mathbb{R}^d$:

$$f(x) = w^T x$$
Linear Regression

- Assume \( f(\cdot) \) is a **linear function** parameterized by \( \mathbf{w} \in \mathbb{R}^d \):
  \[
  f(\mathbf{x}) = \mathbf{w}^T \mathbf{x}
  \]

- Training: compute the model \( \mathbf{w} \) such that \( \mathbf{w}^T \mathbf{x}_i \approx y_i \) for all \( i \)
Linear Regression

- Assume $f(\cdot)$ is a linear function parameterized by $\mathbf{w} \in \mathbb{R}^d$:
  \[ f(x) = \mathbf{w}^T x \]

- Training: compute the model $\mathbf{w}$ such that $\mathbf{w}^T \mathbf{x}_i \approx y_i$ for all $i$

- Equivalent to solving
  \[ \mathbf{w}^* = \text{argmin}_{\mathbf{w} \in \mathbb{R}^d} \sum_{i=1}^{n} (\mathbf{w}^T \mathbf{x}_i - y_i)^2 \]
Assume $f(\cdot)$ is a linear function parameterized by $\mathbf{w} \in \mathbb{R}^d$:

$$f(x) = \mathbf{w}^T x$$

Training: compute the model $\mathbf{w}$ such that $\mathbf{w}^T x_i \approx y_i$ for all $i$

Equivalent to solving

$$\mathbf{w}^* = \arg\min_{\mathbf{w} \in \mathbb{R}^d} \sum_{i=1}^{n} (\mathbf{w}^T x_i - y_i)^2$$

Prediction: given a testing sample $\tilde{x}$, the prediction value is $\mathbf{w}^T \tilde{x}$
Assume the data is generated from the probability model:

\[ y_i \sim \mathbf{w}^T \mathbf{x}_i + \epsilon_i, \quad \epsilon_i \sim \mathcal{N}(0, 1) \]

Maximum likelihood estimator:

\[
\mathbf{w}^* = \arg\max_{\mathbf{w}} \log P(y_1, \ldots, y_n \mid \mathbf{x}_1, \ldots, \mathbf{x}_n, \mathbf{w})
= \arg\max_{\mathbf{w}} \sum_{i=1}^{n} \log P(y_i \mid \mathbf{x}_i, \mathbf{w})
= \arg\max_{\mathbf{w}} \sum_{i=1}^{n} \log \left( \frac{1}{\sqrt{2\pi}} e^{-\frac{(\mathbf{w}^T \mathbf{x}_i - y_i)^2}{2}} \right)
= \arg\max_{\mathbf{w}} \sum_{i=1}^{n} -\frac{1}{2} (\mathbf{w}^T \mathbf{x}_i - y_i)^2 + \text{constant}
= \arg\min_{\mathbf{w}} \sum_{i=1}^{n} (\mathbf{w}^T \mathbf{x}_i - y_i)^2
Linear Regression: written as a matrix form

- Linear regression: \( \mathbf{w}^* = \arg\min_{\mathbf{w} \in \mathbb{R}^d} \sum_{i=1}^{n} (\mathbf{w}^T \mathbf{x}_i - y_i)^2 \)
- Matrix form: let \( \mathbf{X} \in \mathbb{R}^{n \times d} \) be the matrix where the \( i \)-th row is \( \mathbf{x}_i \), \( \mathbf{y} = [y_1, \ldots, y_n]^T \), then linear regression can be written as

\[
\mathbf{w}^* = \arg\min_{\mathbf{w} \in \mathbb{R}^d} \| \mathbf{X}\mathbf{w} - \mathbf{y} \|^2_2
\]
Solving Linear Regression

- Minimize the sum of squared error $J(w)$
  
  $$J(w) = \frac{1}{2} \| Xw - y \|^2$$
  
  $$= \frac{1}{2} (Xw - y)^T (Xw - y)$$
  
  $$= \frac{1}{2} w^T X^T Xw - y^T Xw + \frac{1}{2} y^T y$$

- Derivative: $\frac{\partial}{\partial w} J(w) = X^T Xw - X^T y$

- Setting the derivative equal to zero gives the normal equation

  $$X^T Xw^* = X^T y$$

- Therefore, $w^* = (X^T X)^{-1} X^T y$
Solving Linear Regression

- Minimize the sum of squared error \( J(w) \)

\[
J(w) = \frac{1}{2} \| Xw - y \|^2
\]

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

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

- Derivative: \( \frac{\partial}{\partial w} J(w) = X^T Xw - X^T y \)
- Setting the derivative equal to zero gives the normal equation

\[
X^T Xw^* = X^T y
\]

- Therefore, \( w^* = (X^T X)^{-1} X^T y \)

but \( X^T X \) may be non-invertible. (Will discuss after few classes)
Regularized Linear Regression
Overfitting

- **Overfitting**: the model has low training error but high prediction error.
- Using too many features can lead to overfitting
Regularization to Avoid Overfitting

- Enforce the solution to have low L2-norm:
  \[
  \arg\min_w \sum_{i=1}^{n} \|w^T x_i - y_i\|^2 \text{ s.t. } \|w\|^2 \leq K
  \]

- Equivalent to the following problem with some \( \lambda \)
  \[
  \arg\min_w \sum_{i=1}^{n} \|w^T x_i - y_i\|^2 + \lambda \|w\|^2
  \]
Regularized Linear Regression:

- Regularized Linear Regression:
  \[
  \arg\min_w \|Xw - y\|^2 + R(w)
  \]

  $R(w)$: regularization

- Ridge Regression ($\ell_2$ regularization):
  \[
  \arg\min_w \|Xw - y\|^2 + \lambda \|w\|^2
  \]
Ridge Regression

- Ridge regression: \( \arg\min_{\mathbf{w} \in \mathbb{R}^d} J(\mathbf{w}) \)
  \[
  J(\mathbf{w}) = \frac{1}{2} \| \mathbf{X} \mathbf{w} - \mathbf{y} \|^2 + \frac{\lambda}{2} \| \mathbf{w} \|^2
  \]

- Closed form solution: optimal solution \( \mathbf{w}^* \) satisfies \( \nabla J(\mathbf{w}^*) = 0 \):

  \[
  \mathbf{X}^T \mathbf{X} \mathbf{w}^* - \mathbf{X}^T \mathbf{y} + \lambda \mathbf{w}^* = 0
  \]

  \[
  (\mathbf{X}^T \mathbf{X} + \lambda \mathbf{I}) \mathbf{w}^* = \mathbf{X}^T \mathbf{y}
  \]
Ridge Regression

- Ridge regression: \( \arg \min_{\mathbf{w} \in \mathbb{R}^d} J(\mathbf{w}) \), where
  \[
  J(\mathbf{w}) = \frac{1}{2} \| \mathbf{Xw} - \mathbf{y} \|^2 + \frac{\lambda}{2} \| \mathbf{w} \|^2
  \]

- Closed form solution: optimal solution \( \mathbf{w}^* \) satisfies \( \nabla J(\mathbf{w}^*) = 0 \):
  \[
  \mathbf{X}^T \mathbf{X} \mathbf{w}^* - \mathbf{X}^T \mathbf{y} + \lambda \mathbf{w}^* = 0
  
  (\mathbf{X}^T \mathbf{X} + \lambda \mathbf{I}) \mathbf{w}^* = \mathbf{X}^T \mathbf{y}
  \]

- Optimal solution: \( \mathbf{w}^* = (\mathbf{X}^T \mathbf{X} + \lambda \mathbf{I})^{-1} \mathbf{X}^T \mathbf{y} \)
Ridge Regression

- Ridge regression: \( \text{argmin}_{w \in \mathbb{R}^d} \frac{1}{2} \|Xw - y\|^2 + \frac{\lambda}{2} \|w\|^2 \)

- Closed form solution: optimal solution \( w^* \) satisfies \( \nabla J(w^*) = 0 \):

\[
X^T X w^* - X^T y + \lambda w^* = 0 \\
(X^T X + \lambda I) w^* = X^T y
\]

- Optimal solution: \( w^* = (X^T X + \lambda I)^{-1} X^T y \)

- Inverse always exists because \( X^T X + \lambda I \) is positive definite
Time Complexity

- When $X$ is dense:
  - Closed form solution requires $O(nd^2 + d^3)$ if $X$ is dense
  - Efficient if $d$ is very small
  - Runs forever when $d > 100,000$

- Typical case for big data applications:
  - $X \in \mathbb{R}^{n \times d}$ is sparse with large $n$ and large $d$
  - How can we solve the problem?
Time Complexity

- When $X$ is dense:
  - Closed form solution requires $O(nd^2 + d^3)$ if $X$ is dense
  - Efficient if $d$ is very small
  - Runs forever when $d > 100,000$

- Typical case for big data applications:
  - $X \in \mathbb{R}^{n \times d}$ is sparse with large $n$ and large $d$
  - How can we solve the problem?
    - Iterative algorithms for optimization (next class)
Logistic Regression
Binary Classification

- Input: training data $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n \in \mathbb{R}^d$ and corresponding outputs $y_1, y_2, \ldots, y_n \in \{+1, -1\}$
- Training: compute a function $f$ such that $\text{sign}(f(\mathbf{x}_i)) \approx y_i$ for all $i$
- Prediction: given a testing sample $\tilde{\mathbf{x}}$, predict the output as $\text{sign}(f(\tilde{\mathbf{x}}))$
Logistic Regression

- Assume linear scoring function: $s = f(x) = w^T x$
Logistic Regression

- Assume linear scoring function: \( s = f(x) = \mathbf{w}^T \mathbf{x} \)
- Logistic hypothesis:

\[
P(y = 1 \mid \mathbf{x}) = \theta(\mathbf{w}^T \mathbf{x}),
\]

where \( \theta(s) = \frac{e^s}{1+e^s} = \frac{1}{1+e^{-s}} \)
Error measure: likelihood

- Likelihood of $\mathcal{D} = (x_1, y_1), \cdots, (x_N, y_N)$:

$$\prod_{n=1}^{N} P(y_n \mid x_n)$$
Error measure: likelihood

- Likelihood of $\mathcal{D} = (x_1, y_1), \cdots, (x_N, y_N)$:

$$\prod_{n=1}^{N} P(y_n \mid x_n)$$

- $P(y \mid x) = \begin{cases} 
\theta(w^T x) & \text{for } y = +1 \\
1 - \theta(w^T x) = \theta(-w^T x) & \text{for } y = -1 
\end{cases}$

$\Rightarrow P(y \mid x) = \theta(y w^T x)$
Error measure: likelihood

- Likelihood of $\mathcal{D} = (\mathbf{x}_1, y_1), \cdots, (\mathbf{x}_N, y_N)$:

$$\Pi_{n=1}^N P(y_n \mid \mathbf{x}_n)$$

- $P(y \mid \mathbf{x}) = \begin{cases} 
\theta(\mathbf{w}^T \mathbf{x}) & \text{for } y = +1 \\
1 - \theta(\mathbf{w}^T \mathbf{x}) = \theta(-\mathbf{w}^T \mathbf{x}) & \text{for } y = -1 
\end{cases}$

$\Rightarrow P(y \mid \mathbf{x}) = \theta(y \mathbf{w}^T \mathbf{x})$

Likelihood: $\prod_{n=1}^N P(y_n \mid \mathbf{x}_n) = \prod_{n=1}^N \theta(y_n \mathbf{w}^T \mathbf{x}_n)$
Maximizing the likelihood

Find \( \mathbf{w} \) to maximize the likelihood!

\[
\begin{align*}
\max_{\mathbf{w}} \prod_{n=1}^{N} \theta(y_n \mathbf{w}^T \mathbf{x}_n) \\
\iff \max_{\mathbf{w}} \log(\prod_{n=1}^{N} \theta(y_n \mathbf{w}^T \mathbf{x}_n)) \\
\iff \min_{\mathbf{w}} - \log(\prod_{n=1}^{N} \theta(y_n \mathbf{w}^T \mathbf{x}_n)) \\
\iff \min_{\mathbf{w}} - \sum_{n=1}^{N} \log(\theta(y_n \mathbf{w}^T \mathbf{x}_n)) \\
\iff \min_{\mathbf{w}} \sum_{n=1}^{N} \log\left(\frac{1}{\theta(y_n \mathbf{w}^T \mathbf{x}_n)}\right) \\
\iff \min_{\mathbf{w}} \sum_{n=1}^{N} \log(1 + e^{-y_n \mathbf{w}^T \mathbf{x}_n})
\end{align*}
\]
Most linear ML algorithms follow

$$\min_w \frac{1}{N} \sum_{n=1}^{N} \text{loss}(w^T x_n, y_n)$$

- Linear regression: \(\text{loss}(h(x_n), y_n) = (w^T x_n - y_n)^2\)
- Logistic regression: \(\text{loss}(h(x_n), y_n) = \log(1 + e^{-y_n w^T x_n})\)
Empirical Risk Minimization (general)

- Assume $f_W(x)$ is the decision function to be learned ($W$ is the parameters of the function)
- General empirical risk minimization:
  \[
  \min_W \frac{1}{N} \sum_{n=1}^{N} \text{loss}(f_W(x_n), y_n)
  \]
- Example: Neural network ($f_W(\cdot)$ is the network)
Questions?