Acknowledgement:
Many of these slides are derived from Tom Mitchell, Pascal Poupart, Pieter Abbeel, Eric Eaton, Carlos Guestrin, William Cohen, and Andrew Moore.
Regression

- So far, we’ve been interested in learning $P(Y|X)$ where $Y$ has discrete values (called ‘classification’)

- What if $Y$ is continuous? (called ‘regression’)
  - predict snow/rainfall from current and past weather features
  - predict stock price from current and past market conditions
  - predict weight from gender, height, age, ...
Regression: problem setting

- Wish to learn \( f: X \rightarrow Y \) where \( Y \) is real-valued, given training data \( \{ <X^1, Y^1>, \ldots, <X^n, Y^n> \} \)

- Approach:
  - Choose some parameterized form for \( P(Y | X, \theta) \) where \( \theta \) is the vector of parameters
  - Estimate \( \theta \) using MLE or MAP estimation
Choose parameterized form for $p(Y | X, \theta)$

- Assume $Y$ is some deterministic $f(X)$, plus random noise
  - $y = f(x) + \varepsilon$ where $\varepsilon \sim \mathcal{N}(0, \sigma)$
- Therefore, $Y$ is a random variable that follows the distribution
  - $p(y | x) = \mathcal{N}(f(x), \sigma)$
- And the expected value of $y$ for any given $x$ is $f(x)$
Consider linear regression

- $p(y | x) = \mathcal{N}(f(x), \sigma)$
  \[ f(x) = w_0 + w_1 x \]

- Assume $f(x)$ is a linear function of $x$
  \[ p(y | x) = \mathcal{N}(w_0 + w_1 x, \sigma) \]

- $\mathbb{E}(y | x) = w_0 + w_1 x$
Consider linear regression

- $p(y \mid x) = \mathcal{N}(f(x), \sigma)$

- Assume $f(x)$ is a linear function of $x$
  
  i.e., $y = w_0 + w_1x$

  - $p(y \mid x) = \mathcal{N}(w_0 + w_1x, \sigma)$
  - $\mathbb{E}(y \mid x) = w_0 + w_1x$

- Note: to make our parameters explicit, let’s write
  
  - $W = \langle w_0, w_1 \rangle$
  - $p(y \mid x, W) = \mathcal{N}(w_0 + w_1x, \sigma)$
Training linear regression

- $p(y|x) = \mathcal{N}(f(x), \sigma)$

- How can we learn $W$ from data?

- Learn $W$ using Maximum Conditional Likelihood Estimation!
  - $W_{MCLE} = \arg \max_W \prod_l P(Y^l | X^l, W)$
  - $W_{MCLE} = \arg \max_W \sum_l \ln p(y^l | x^l, W)$
  - Where $P(y|x, W) = \frac{1}{\sigma \sqrt{2\pi}} e^{-\frac{1}{2} \left( \frac{y^l - f(x, W)}{\sigma} \right)^2}$
**MCLE derivation**

\[
W_{MCLE} = \arg \max_W \sum_l \ln p(y^l \mid x^l, W)
\]

\[
P(y \mid x, W) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{y^l - f(x, W)}{\sigma}\right)^2}
\]

or

\[
P(y \mid x, W) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{y^l - (w_0 + w_1 x^l)}{\sigma}\right)^2}
\]

\[
W_{MCLE} = \arg \max_W \sum_l \ln \frac{1}{\sigma\sqrt{2\pi}} + C - \frac{1}{2} \left(\frac{y^l - (w_0 + w_1 x^l)^2}{\sigma^2}\right)
\]

\[
\text{maximization} = \arg \max_W \sum_l \frac{1}{2\sigma^2} \left(y^l - (w_0 + w_1 x^l)^2\right)
\]

\[
\text{minimization} = \arg \min_W \sum_l \frac{1}{2\sigma^2} \left(y^l - (w_0 + w_1 x^l)^2\right)
\]

\[
= \arg \min_W \sum_l \left(y^l - (w_0 + w_1 x^l)^2\right)
\]

\[
f(x) = w_0 + w_1 x
\]
Training linear regression

- Learn $W$ using Maximum Conditional Likelihood Estimation:
  
  $$W_{MCLE} = \arg \min_W \sum_l (y - f(x, W))^2$$

- This corresponds to minimizing sum of squared errors (often used in “curve fitting”)

- How to perform the minimization in order to choose optimal $W = <w_0, w_1>$?
Gradient descent

\[ E(\vec{W}) \]

\[ \nabla E(\vec{w}) = \left[ \frac{\partial E}{\partial w_0}, \frac{\partial E}{\partial w_1}, \ldots, \frac{\partial E}{\partial w_n} \right] \]

\[ \text{Training Rule: } \vec{w}^{(i+1)} \leftarrow \vec{w}^{(i)} - \eta \nabla E(\vec{w}) \]

\[ \Delta w_i = -\eta \frac{\partial E}{\partial w_i} \]
Minimizing squared error: gradient descent

\[
\frac{\partial E}{\partial w_1} = \sum_l 2(y^l - (w_o + w_1x^l))(-x^l) = -2 \sum_l (y^l - (w_o + w_1x^l))(x^l)
\]

\[
\frac{\partial E}{\partial w_0} = -2 \sum_l (y^l - (w_o + w_1x^l))
\]
Minimizing squared error: gradient descent

\[
\frac{\partial E}{\partial w_1} = -2 \sum_l (y^l - (w_o + w_1x^l))(x^l)
\]
\[
\frac{\partial E}{\partial w_0} = -2 \sum_l (y^l - (w_o + w_1x^l))
\]

Update rule:

\[
\omega_1 \leftarrow \omega_1 - \eta \left[ -2 \sum_l x^l (y^l - (w_o + w_1x^l)) \right]
\]
\[
\omega_1 \leftarrow \omega_1 + 2\eta \sum_l x^l (y^l - (w_o + w_1x^l))
\]
\[
\omega_0 \leftarrow \omega_0 + 2\eta \sum_l (y^l - (w_o + w_1x^l))
\]
Linear regression more generally

\[ p(y \mid x) = \mathcal{N}(f(x), \sigma I) \quad \vec{X} = < x_1, x_2, \ldots, x_n > \]

\[ f(x) = w_0 + \sum_{i=1}^{n} w_i x_i \]

\[ p(y \mid x) = \mathcal{N}(w_0 + \sum_{i=1}^{n} w_i x_i, \sigma I) \]

\[ \mathbb{E}(y \mid x) = w_0 + \sum_{i=1}^{n} w_i x_i \]

\[ p(y \mid x, W) = \mathcal{N}(w_0 + \sum_{i=1}^{n} w_i x_i, \sigma I) \quad \vec{W} = < w_0, w_1, \ldots, w_n > \]
Minimizing squared error more generally: gradient descent

\[ f(x) = w_0 + \sum_{j=1}^{n} w_j x_j \]

- **Gradient descent algorithm**: iterate until change < \( \varepsilon \)
  
  \[ w_i \leftarrow w_i + 2\eta \sum_{l} X^l_i (Y^l - (w_0 + \sum_{j=1}^{n} w_j x_j)) \]

- **assume** \( X_0 = 1 \) for \( w_0 \)
How about MAP estimation?

- \( W_{MAP} = \arg \max_W (-c \sum_l w_i^2) + \sum_l \ln P(Y^l | X^l, W) \)

- Called a "regularization" term
- Helps reduce overfitting, especially for sparse data situations
- Keeps weights near zero with prior \( W \sim \mathcal{N}(0, \sigma I) \), or whatever the prior suggests
Demo Time 😊

https://lukaszkujawa.github.io/gradient-descent-descent.html
Summary of regression

- Assuming $p(y|x, W) = \mathcal{N}(w_0 + w_1x, \sigma)$

- MLE corresponds to minimizing sum of squared errors
- MAP estimate minimizes SSE plus sum of squared weights
- Again, learning is an optimization problem once we choose our objective function
  - maximize data likelihood
  - maximize posterior prob of $W$
- Again, we can use gradient descent as a general learning algorithm

- Be careful about outliers while performing regression

- **NOTE**: Almost nothing here required that $f(x)$ be linear in $x$