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

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

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

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

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

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

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

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

- How can we learn \( W \) from data?

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

- How to perform the minimization in order to choose optimal
  \[ W = < w_0, w_1 > \]
Gradient descent

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

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

- i.e. \( \Delta w_i = -\eta \frac{\partial E}{\partial w_i} \)
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:
Linear regression more generally

- \( p(y \mid x) = \mathcal{N}(f(x), \sigma I) \quad \vec{X} = \langle x_1, x_2, \ldots, x_n \rangle \)

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

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

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

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

- **Gradient descent algorithm**: iterate until change < \( \varepsilon \)
  
  \[
  \forall \ i \ \text{repeat} \quad w_i \leftarrow w_i + 2\eta \sum_l X_l^i (Y_l - (w_0 + \sum_{j=1}^n w_j x_j))
  \]
  
  - assume \( X_0 = 1 \) for \( w_0 \)
How about MAP estimation?

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

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

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

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

- 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 $x$