today:  - logistic regression
         - numerical optimization
         IRLS

Logistic regression model:
\[
Y = \mathcal{N}(0, 1)^2
\]
\[
p(y=1|x) = \sigma(w^Tx)
\]
\[
p(y=0|x) = 1 - \sigma(w^Tx) = \sigma(-w^Tx)
\]
\[\text{i.e. } Y | X=x \text{ is Bernoulli } (\sigma(w^Tx))\]
\[
p(y|x) = \sigma(w^Tx)^y \sigma(-w^Tx)^{1-y}
\]

Given \( (x_i, y_i)_{i=1}^n \), maximum conditional likelihood:
\[
l(w) = \sum_{i=1}^n \log p(y_i|x_i; w) = \sum_{i=1}^n \left[ y_i \log \sigma(w^Tx_i) + (1-y_i) \log \sigma(-w^Tx_i) \right]
\]

derivative
\[
\frac{\partial}{\partial w} (w) = \sigma(w^Tx) \sigma(-w^Tx)
\]
\[
\nabla_w \sigma(w^Tx_i) = x_i \left[ \sigma(w^Tx_i) \sigma(-w^Tx_i) \right] \quad \text{Let } \nu_i = \omega^T x_i
\]
\[
\nabla l(w) = \sum_{i=1}^n x_i \left[ y_i \frac{\partial}{\partial \nu_i} \sigma(-\nu_i) - (1-y_i) \frac{\partial}{\partial \nu_i} \sigma(\nu_i) \right] \left[ y_i \left[ \sigma(-\nu_i) + \sigma(\nu_i) \right] - \sigma(\nu_i) \right] - \sigma(\nu_i)
\]
\[ \nabla l(w) = \sum_{i=1}^{n} x_i \left[ y_i - \sigma(w^T x_i) \right] \]

solve for \( \nabla l(w) = 0 \) \( \Rightarrow \) need to solve a transcendental eq.

because \( \frac{1}{1 + e^{w^T x_i}} \) need to solve for \( w \)

contrasted to least square regression:
\[ \nabla l(w) = \sum_{i=1}^{n} x_i \left[ y_i - \sigma(w^T x_i) \right] \]
linear in \( w \)

\underline{numerical optimization}

want to minimize \( f(w) \)
\[ \min_{w \in \mathbb{R}^q} f(w) \]

1) \underline{gradient descent} (1st order method)

\[ w_{t+1} = w_t - \eta \nabla f(w_t) \]

step-size
step-size rules:

a) constant step-size \( \alpha_t = \frac{1}{L} \) Lipschitz constant of \( \nabla f \)
\[ \| \nabla f(w_t) - \nabla f(w) \| \leq L \| w - w_t \| \]

b) decreasing step-size rule: \( \alpha_t = \frac{\gamma}{t} \) constant
usually want: \( \lim_{t \to \infty} \alpha_t = 0 \)
\[ \gamma > 0 \]
direction for update (e.g. \( \nabla f(w_t) \))

c) choose \( \alpha_t \) by "line search" : \( \min_{\delta \in \mathbb{R}} \{ f(w_t + \alpha t \delta) \} \)
\[ \text{costly in general} \]
instead do approximate search (e.g. Armijo line search)
(see Boyd's book)

2) Newton's method (2nd order method)
motivation: minimizing a quadratic approximation:

Taylor expansion:
\[ f(w) = f(w_t) + (\nabla f(w_t))^T (w - w_t) + \frac{1}{2} (w - w_t)^T H(w_t) (w - w_t) \]
\[ + O(\| w - w_t \|^3) \]
\[ = Q_t(w) + O(\| w - w_t \|^3) \]
\[ \text{Taylor's remainder} \]
(\omega_n \rightarrow \text{minimizing } Q(t, \omega))

\nabla_{\omega} Q(t, \omega) = 0

\nabla f(\omega_t) + H(\omega_t)(\omega - \omega_t) = 0

\Rightarrow \omega_{t+1} = \omega_t - H^{-1}(\omega_t) \nabla f(\omega_t)

- Inverse Hessian \rightarrow O(d^3) \text{ time to compute in general}
- O(d^2) \text{ space}

\text{Newton's update}

\text{Damped Newton: you add a stepsize to stabilize Newton's method}

\omega_{t+1} = \omega_t - \alpha H^{-1}(\omega_t) \nabla f(\omega_t)

\text{why Newton's method?}

- much faster convergence in $\# \text{ of iterations }$ vs gradient descent
- affine invariant $\rightarrow$ invariant by rescaling of variables
- Newton is transforming space

\text{Good sets for } f(\omega) \text{ using Hessian to make it...}
\[ z = \mathbf{H}^{-1} \alpha \]
\[ \nabla f(z) = \mathbf{H}^{-1} \nabla f(\alpha) \]
\[ \frac{1}{2} x^T \mathbf{H} x = c \]
\[ \frac{1}{2} x^T \mathbf{P}^T \mathbf{P} x < c \]

Look at "quadratic forms"

(sidenote: implicit regularization properties of optimization method)

Newton's method for logistic regression & IRLS

\[ \nabla l(\mathbf{w}) = \sum_{i=1}^{n} x_i \left[ y_i - \sigma(\mathbf{w}^T x_i) \right] \]
\[ H(l(\mathbf{w})) = \sum_{i=1}^{n} x_i x_i^T \sigma(\mathbf{w}^T x_i) \sigma(\mathbf{w}^T x_i) \]
\[ \nabla^T H \nabla = -\sum_{i=1}^{n} \left( \frac{(\nabla^2 \sigma(-\mathbf{w}^T x_i)) (\nabla^2 \sigma(-\mathbf{w}^T x_i))}{(\nabla \sigma(\mathbf{w}^T x_i))^2} \right) \geq 0 \]

ie. \( \nabla^T H \nabla \geq 0 \)

\[ \Rightarrow \ H \geq 0 \]

ie. concave function

notation:
\[ \mathbf{X} = \left( \begin{array}{c} -z_i^T \\ \vdots \\ -z_1^T \end{array} \right) \]

let \( \mu_i \triangleq \sigma(\mathbf{w}^T x_i) \in [0,1] \)
\[ \nu_i \triangleq \sigma(\omega_i) \in [0,1] \]
\[ \nabla \ell(w) = \sum_{i=1}^{n} x_i [y_i - \nu_i] = X^T (y - \mu) \]
Hessian: \[ = -\sum_{i=1}^{n} x_i x_i^T \nu_i (1 - \nu_i) = -X^T D \nu X \]
where \( D_{ii} = \nu_i (1 - \nu_i) \)

[Note: \( D \) depends]

Newton's update (here Newton's updates are maximizing log-likelihood, \( \frac{d}{d\nu} \ell \) concave)

Newton's update:
\[ w_{k+1} = w_k - (X^T D_x X)^{-1} X^T (y - \mu w) \]
\[ = (X^T D_x X)^{-1} \left[ (X^T D_x X) w_k + X^T (y - \mu w) \right] \]
\[ w_{k+1} = (X^T D_x X)^{-1} \left( X^T D_x z_k \right) \]
where \( z_k \triangleq X w_k + D^{-1}_x (y - \mu w) \)

This is a solution to "weighted least squares problem"
\[ \min_{w} \sum_{i=1}^{n} D_{ii}^2 (z_i - x_i w)^2 \]

Compare with Gaussian noise model for L.S.
\[ \sum_{i=1}^{n} \frac{(y_i - x_i w)^2}{\sigma_i^2} \]
Neutron's method for logistic regression

= Iterative Reweighted Least Square (IRLS)

2 comments for assignment:

\( \| \nabla f(w) \|_2 \leq E \)

2) to compute \( A^T \) \( A \) \( \backslash \) \( V \)

Big data logistic regression:

- cannot do \( O(d^3) \) or \( O(d^2) \) operations \( \Rightarrow \) First order methods
- if \( n \) is huge, you cannot do batch method

\( \nabla f(w) = \frac{1}{n} \sum_{i=1}^{n} \nabla f_i(w) \quad O(nd) \) time

instead you "incremental gradient methods"

Eq. Stochastic gradient descent (SGD):

\( w_{t+1} = w_t - \alpha_t \nabla f_i(w_t) \quad O(d) \) time

where \( \alpha_t \) is picked randomly

SGD \( \Rightarrow \) cheap updates, but slower convergence per iteration
SAG: stochastic averaged gradient

C.D. \[ w_{t+1} = w_t - \alpha_t \frac{1}{n} \sum_{i=1}^{n} \nabla f_i(w_t) \]

SAG: \[ w_{t+1} = w_t - \alpha_t \frac{1}{n} \sum_{i=1}^{n} \nabla f_i(w_t) \text{ memory} \]

where \( \nabla f_i(w_t) \) at each \( t \), update only one \( V_{i,t} \)

SA(2014): \[ w_{t+1} = w_t - \alpha_t \left( \nabla f_i_t(w_t) + \frac{1}{n} \sum_{i=1}^{n} \nabla f_i_t(w_t) \right) \text{ variance reducing correction} \]

(default method for logistic regression in Scikit-learn)