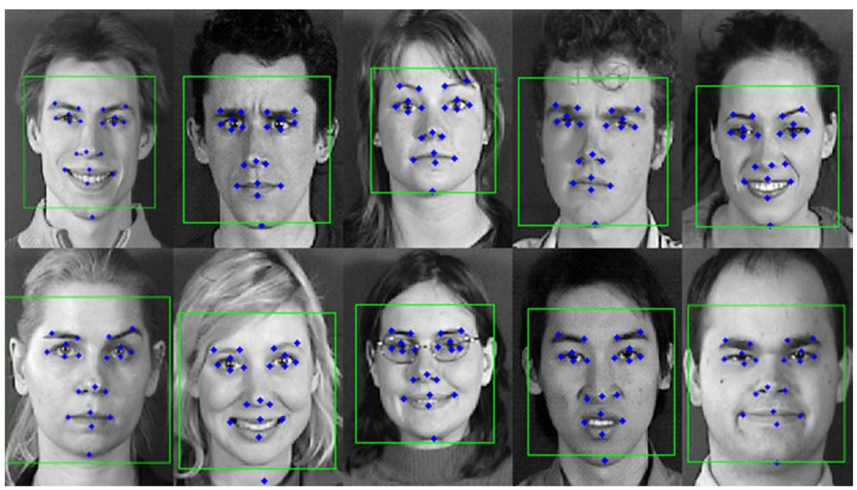
Learning from Data
7. Classification: Logistic Regression

Spring 2017

Classification

- Supervised (preditive) learning: given examples with labels, predict labels for all unseen examples
  - Classification:
    - label = category,
    - $\mathbf{y} \in \mathcal{Y} = \{1, \ldots, m\}$, m = number of classes
    - $\ell(\mathbf{x}, y, h) = 1(h(\mathbf{x}) \neq y)$, Risk = $P(Y \neq h(\mathbf{X})) = P(\text{Error})$



$\mathbf{x}$ = facial geometry features
y = gender label

Logistic Regression for Classification

- Discriminative learning: only $p(y|\mathbf{x})$ estimated
- Parametric:

$$p(y|\mathbf{x},\theta) = \frac{e^{\mathbf{w}_y^\top \mathbf{x}}}{\sum_{k=1}^m e^{\mathbf{w}_k^\top \mathbf{x}}} = \mathrm{softmax}(y|\theta^\top \mathbf{x}),$$

$$\theta = [\mathbf{w}_1, \ldots, \mathbf{w}_m] \in \mathbb{R}^{d\times m}, y \in \{1, \ldots, m\}$$

- Note:

$\theta = [\mathbf{w}_1, \ldots, \mathbf{w}_m]$ and $\tilde{\theta} = [\mathbf{w}_1 - \mathbf{w}_m, \ldots, \mathbf{w}_m - \mathbf{w}_m]$ are two different but equivalent parameterizations of the exact same model, i.e., a model with θ and another with $\tilde{\theta}$ are indistinguishable

- To ensure identifiability (uniqueness), we will assume that $\mathbf{w}_m = \mathbf{0} \Rightarrow$ there are only $d \times (m-1)$ scalar parameters. We will relax this in regularized logistic reg.

Logistic Regression for Classification

- Revised expressions:
- Parametric:

$$p(y|\mathbf{x},\theta) = \frac{e^{\mathbf{w}_y^\top \mathbf{x}}}{\sum_{k=1}^{m} e^{\mathbf{w}_k^\top \mathbf{x}}} = \mathrm{softmax}(y|\theta^\top \mathbf{x}),$$

$$\theta = [\mathbf{w}_1, \ldots, \mathbf{w}_{m-1}, \mathbf{w}_m = \mathbf{0}] \in \mathbb{R}^{d\times m}$$

- MPE/MAP classifier:

$$\begin{aligned} h_{\mathrm{MPE}}(\mathbf{x}) &= \arg \max_{y\in\{1,\ldots,m\}} p(y|\mathbf{x},\theta) \\ &= \arg \max_{y\in\{1,\ldots,m\}} \mathbf{w}_y^\top \mathbf{x} \end{aligned}$$

  – A linear classifier as in LDA, but the weights are learned by maximizing the conditional likelihood

Estimation of parameters

- Frequentist Discriminative Learning of θ:

$$
\begin{aligned}
\widehat{\theta}_{\mathrm{ML}}(\mathcal{D}) &= \arg\max_{\theta} \prod_{j=1}^{n} p(y_j|\mathbf{x}_j, \theta) \\
&= \arg\max_{\theta} \sum_{j=1}^{n} \ln\left(\frac{e^{\mathbf{w}_{y_j}^{\top}\mathbf{x}_j}}{\sum_{k=1}^{m} e^{\mathbf{w}_k^{\top}\mathbf{x}_j}}\right) \\
&= \arg\min_{\theta} \sum_{j=1}^{n} \left[\ln\left(\sum_{k=1}^{m} e^{\mathbf{w}_k^{\top}\mathbf{x}_j}\right) - \mathbf{w}_{y_j}^{\top}\mathbf{x}_j\right] \\
&= \arg\min_{\theta} \left\{\sum_{j=1}^{n} \ln\left(\sum_{k=1}^{m} e^{\mathbf{w}_k^{\top}\mathbf{x}_j}\right) - \sum_{k=1}^{m} \mathbf{w}_k^{\top}\left(\sum_{j=1}^{n} 1(y_j = k)\mathbf{x}_j\right)\right\}
\end{aligned}
$$

Negative Log-Likelihood (NLL)

- $$\mathrm{NLL}(\mathbf{w}_1,\ldots,\mathbf{w}_m) = \sum_{j=1}^{n} \ln\left(\sum_{k=1}^{m} e^{\mathbf{w}_k^\top \mathbf{x}_j}\right) - \sum_{k=1}^{m} \mathbf{w}_k^\top \left(\sum_{j=1}^{n} 1(y_j = k)\mathbf{x}_j\right)$$
- Gradient of NLL w.r.t. $\mathbf{w}_y, y = 1, \ldots, m-1$ is given by:
$$\nabla_{\mathbf{w}_y}\mathrm{NLL}(\mathbf{w}_1,\ldots,\mathbf{w}_m) = \sum_{j=1}^{n}\left(\underbrace{\frac{e^{\mathbf{w}_y^\top \mathbf{x}_j}}{\sum_{k=1}^{m} e^{\mathbf{w}_k^\top \mathbf{x}_j}}}_{p(y|\mathbf{x}_j,\theta)} - 1(y_j = y)\right)\mathbf{x}_j$$
- The Hessian $\nabla_\theta^2\mathrm{NLL}(\theta)$ can be shown to be a positive semi-definite matrix (positive definite if $w_m \equiv 0$)
- $\Rightarrow$ NLL(θ) is a convex differentiable function of θ (strictly convex if $w_m \equiv 0$)
- $\Rightarrow$ local minimum of NLL(θ) = global minimum
- Gradient Descent Algorithm can find a global minimizer (the minimizer is unique, if $w_m \equiv 0$)

Gradient (Steepest) Descent Algorithm

- Initialization: $\theta_0 \equiv \mathbf{w}_1^{(0)}, \dots, \mathbf{w}_{m-1}^{(0)}, \mathbf{w}_m^{(0)} = 0$
- for t = 1, 2, …, until Stopping Criterion, do:
  - Evaluate gradients:
    $$\nabla_{\mathbf{w}_1} \mathrm{NLL}(\theta_t), \dots, \nabla_{\mathbf{w}_{m-1}} \mathrm{NLL}(\theta_t)$$
  - Update weights:
    $$\mathbf{w}_y^{(t+1)} = \mathbf{w}_y^{(t)} - \eta_t \nabla_{\mathbf{w}_y} \mathrm{NLL}(\theta_t), \qquad y = 1, \dots, m-1$$
- endfor

Gradient Descent Algorithm

- Initialization:
  - all zeros
  - random, e.g., iid zero mean spherical Gaussian
- Stopping criterion:
  - Max. # iterations: $t \leq t_{\max}$
  - NLL does not change much: $|\mathrm{NLL}(\theta_{t+1}) - \mathrm{NLL}(\theta_t)| \leq \epsilon$
  - Gradients are almost zero: $\max_{1 \leq y \leq m-1} \left\| \nabla_{\mathbf{w}_y} \mathrm{NLL}(\theta_t) \right\| \leq \epsilon$
- Step-size (learning rate) η_t:
  - Fixed: $\eta_t = \eta, \ \forall t$: too small ➔slow convergence; too large ➔fail to converge
  - Adaptive (line search):

$$\eta_t = \mathrm{argmin}_{0 \leq \eta} \mathrm{NLL}(\theta_t - \eta \nabla \mathrm{NLL}(\theta_t))$$

Gradient Descent Algorithm

- Line search: $\eta_t = \mathrm{argmin}_{0 \leq \eta} \mathrm{NLL}(\theta_t - \eta \nabla \mathrm{NLL}(\theta_t))$
- Scalar optimization problem
- Derivative w.r.t. $\eta = 0 \Rightarrow \eta_t$ satisfies following equation:

$$\underbrace{\nabla^\top \mathrm{NLL}(\theta_t - \eta_t \nabla \mathrm{NLL}(\theta_t))}_{\text{gradient at end of step}} \cdot \underbrace{\nabla \mathrm{NLL}(\theta_t)}_{\text{gradient at beginning of step}} = 0$$

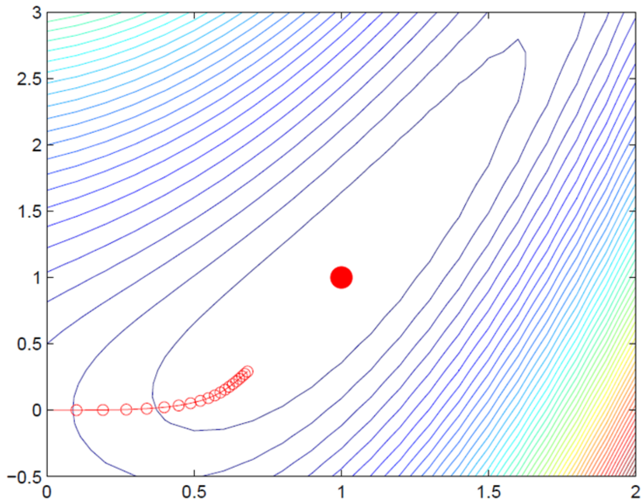
- $\Rightarrow$ either gradient at end of step = 0 $\Rightarrow$ global minimum found
- or gradient at end of step $\perp$ gradient at beginning of step $\Rightarrow$ zig-zag path (see pictures on next slide)

Gradient Descent Algorithm (example)

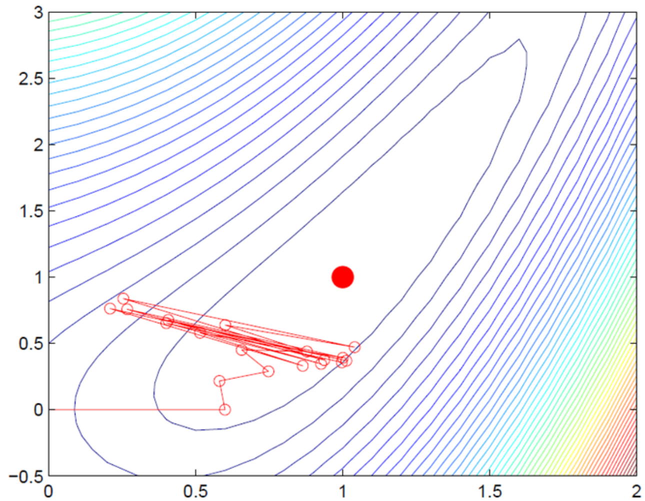
- $f(\theta) = 0.5(\theta_1^2 - \theta_2)^2 + 0.5(\theta_1 - 1)^2$, zero initialization

Fixed step size:

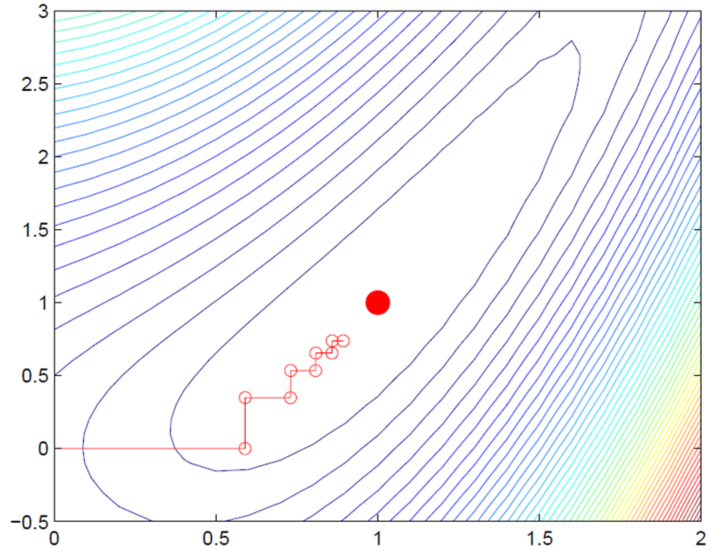
$\eta = 0.1$
(converges slowly)



$\eta = 0.6$
(fails to converge)



Line search:



Other algorithms

- Method of Conjugate Gradients

- Newton's Method
  - Iteratively Reweighted Least Squares (IRLS)
  - requires Hessian computation (expensive)

- Quasi-Newton Method (BFGS: Broyden, Fletcher, Goldfarb, Shanno)

Regularization

- In general, regularization is needed when $n \ll d$
- In logistic regression for classification, regularization may be needed even if $n \gg d$:
  - Suppose that the classes are perfectly linearly separable for some $\theta \Rightarrow$ they are also perfectly separable for $\frac{1}{T}\theta$ for any $T > 0$.
  - Then the likelihood which is given by the softmax function is maximized when $T \rightarrow 0$ which converts the softmax function to a true max function and assigns the maximum probability mass to the training data
  - In other words, NLL is minimized when $\left\|\mathbf{w}_y\right\| \rightarrow \infty$ for all $y < m$. This makes the gradient descent algorithm unstable motivating the need for regularization

$\ell_2$regularization

- IID Gaussian prior on weights (here we don't assume $\mathbf{w}_m = \mathbf{0}$ anymore):

$$\pi(\theta) = \pi(\mathbf{w}_1, \ldots, \mathbf{w}_m) = \prod_{k=1}^{m} \mathcal{N}(\mathbf{0}, \Sigma_{\text{reg}})(\mathbf{w}_k), \quad \Sigma_{\text{reg}} > 0$$

- $\hat{\theta}_{\text{Bayes}}(\mathcal{D}) = \arg\min_{\theta} [-\ln p(\mathcal{D}|\theta) - \ln \pi(\theta)]$

$$= \arg\min_{\theta} f(\theta)$$

$$f(\theta) = \text{NLL}(\theta) + \frac{1}{2}\sum_{k=1}^{m} \mathbf{w}_k^{\text{T}} \Sigma_{\text{reg}}^{-1} \mathbf{w}_k$$

$$\nabla_{\mathbf{w}_y} f(\theta) = \nabla_{\mathbf{w}_y} \text{NLL}(\theta) + \Sigma_{\text{reg}}^{-1} \mathbf{w}_y$$

- $f(\theta)$ is strictly convex and differentiable since for each k, $\mathbf{w}_k^{\text{T}} \Sigma_{\text{reg}}^{-1} \mathbf{w}_k$ is strictly convex and differentiable, and the sum of convex functions and a strictly convex function is strictly convex
- Therefore $f(\theta)$ has a unique global minimum which can be found by the Gradient Descent Algorithm (suitably modified)

Remarks

Naïve Bayes (Generative)	Logistic Regression (Discriminative)
Typically fewer parameters due to independence assumption	
Stronger modeling assumption (independence) ⇒ less robust to model mismatch	Weaker modeling assumptions ⇒ more robust to model mismatch
Converges faster with increasing data n to a less accurate classifier	Converges slower with increasing data n to a more accurate classifier
Easier to learn and update since parameters of each class are optimized separately, but each class has only a fraction of the total number of training examples	Joint optimization: harder to update, but uses all training examples in optimization jointly.

Proof of convexity of NLL

- First we show that the function

$$f(\mathbf{v}) := \ln(\textstyle\sum_{u=1}^{m} e^{v_u}), \mathbf{v} = (v_1, \dots, v_m)^T$$

is convex by showing that its Hessian is positive semi-definite: Let $s(\mathbf{v}) = \sum_{u=1}^{m} e^{v_u}$. Then we have

$$\begin{aligned}
\frac{\partial f}{\partial v_j} &= \frac{e^{v_j}}{s(\mathbf{v})} \\
\Rightarrow \frac{\partial^2 f}{\partial v_j \partial v_k} &= \frac{s(\mathbf{v}) e^{v_j} 1(j=k) - e^{v_j} e^{v_k}}{s^2(\mathbf{v})} \\
\Rightarrow \sum_{j,k} a_j a_k \frac{\partial^2 f}{\partial v_j \partial v_k} &= \frac{1}{s^2(\mathbf{v})} \Big[s(\mathbf{v}) \sum_j a_j e^{v_j} \sum_k a_k 1(j=k) - \sum_{j,k} a_j a_k e^{v_j} e^{v_k} \Big] \\
&= \frac{1}{s^2(\mathbf{v})} \Big[\sum_j e^{v_j} \sum_j a_j^2 e^{v_j} - \big(\sum_j a_j e^{v_j} \big)^2 \Big] \\
&\geq 0 \quad \dots \quad \text{(Cauchy-Schwartz inequality)}
\end{aligned}$$

Proof of convexity of NLL

- If $v_m \equiv 0$, then

$$f(\mathbf{v}) := \ln(\sum_{u=1}^{m} e^{v_u}), \mathbf{v} = (v_1, \dots, v_{m-1}, 0)^T$$

is strictly convex w.r.t. $(v_1, \dots, v_{m-1})^T$ since its Hessian is positive definite: With $s(\mathbf{v}) := 1 + \sum_{u=1}^{m-1} e^{v_u}$,

$$\Rightarrow \sum_{j=1,k=1}^{m-1,m-1} a_j a_k \frac{\partial^2 f}{\partial v_j \partial v_k} = \frac{1}{s^2(\mathbf{v})}\left[s(\mathbf{v}) \sum_{j=1}^{m-1} a_j e^{v_j} \sum_{k=1}^{m-1} a_k 1(j=k) - \sum_{j,k} a_j a_k e^{v_j} e^{v_k} \right]$$

$$= \frac{1}{s^2(\mathbf{v})}\left[s(\mathbf{v}) \sum_j a_j^2 e^{v_j} - \Big(\sum_j a_j e^{v_j}\Big)^2 \right]$$

$$= \frac{1}{s^2(\mathbf{v})}\left[\underbrace{\sum_j a_j^2 e^{v_j}}_{\geq 0, =0 \text{ iff } a_j = 0 \forall j} + \underbrace{\sum_{j=1}^{m-1} e^{v_j} \sum_{j=1}^{m-1} a_j^2 e^{v_j} - \Big(\sum_{j=1}^{m-1} a_j e^{v_j}\Big)^2}_{\geq 0 \text{ by Cauchy-Schwartz inequality}} \right]$$

≥ 0 and is equal to zero iff $a_j = 0$ for all j

Proof of convexity of NLL

- Next, we make a few observations about convex functions:
  - A linear function is a convex function
  - A convex function of a linear function is a convex function
  - The sum of convex functions is a convex function

$$\mathrm{NLL}(\underbrace{[\mathbf{w}_1, \ldots, \mathbf{w}_m]}_{\theta}) = \underbrace{\sum_{j=1}^{n} \underbrace{\ln\left(\sum_{k=1}^{m} e^{\mathbf{w}_k^\top \mathbf{x}_j}\right)}_{\text{convex function of } \underbrace{\theta^\top \mathbf{x}_j}_{\text{linear function of } \theta}}}_{\text{sum of convex functions of } \theta} \underbrace{- \sum_{k=1}^{m} \mathbf{w}_k^\top \left(\sum_{j=1}^{n} 1(y_j = k)\mathbf{x}_j\right)}_{\text{linear function of } \theta}$$

  - If $w_m \equiv 0$, then NLL is a strictly convex function of θ since the log-sum-exp is so and the sum of a strictly convex function and convex functions is strictly convex