Linear Models for Classification

Evgeny Burnaev

Skoltech, Moscow, Russia

Outline

1. Optimal Bayesian Classifier
2. Discriminant Analysis
3. Learning a Classifier
4. Logistic Regression

Problem Statement I

- Let
  - — X be a feature space, Y be a space of labels, e.g. $Y = \{0, 1\}$
  - — $p(\mathbf{x}, y)$ be a joint distribution on $X \times Y$
  - — $S_m = \{(\mathbf{x}_i, y_i)\}_{i=1}^m$ be an i.i.d. sample
- We want to construct an optimal classifier $h : X \to Y$

Problem Statement II

- We assume that we know joint density

$$p(\mathbf{x}, y) = p(\mathbf{x})p(y|\mathbf{x}) = p(y)p(\mathbf{x}|y)$$

  here

  — $p(y)$ is a prior distribution on Y
  — $p(\mathbf{x}|y)$ is a likelihood of a class y
  — $p(y|\mathbf{x})$ is a posterior probability of a class y

- Classifier maximizing posterior probability

$$h(\mathbf{x}) = \arg\max_{y \in Y} p(y|\mathbf{x}) = \arg\max_{y \in Y} p(\mathbf{x}|y)p(y)$$

Probability of Error and Risk

- Classifier $h(\mathbf{x})$ divides X into disjoint domains

$$H_y = \{\mathbf{x} \in X | h(\mathbf{x}) = y\},\ y \in Y$$

- We get error for $(\mathbf{x}, y)$ if $\mathbf{x} \in H_z$, $z \neq y$
- Probability of Error: $P(H_z, y) = \int_{H_z} p(\mathbf{x}, y) d\mathbf{x}$
- Losses: λ_{yz} for all $(y, z) \in Y \times Y$
- Average Risk

$$R(h) = \sum_{y \in Y} \sum_{z \in Y} \lambda_{yz} P(H_z, y)$$

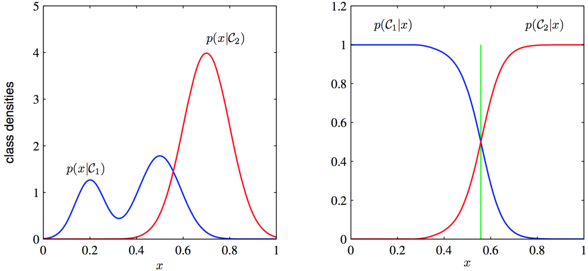
Optimal Bayesian Classifier

- **Theorem**: Optimal Bayesian Classifier $h(\mathbf{x})$, minimizing average risk $R(h)$, has the form

$$h_{\text{opt}}(\mathbf{x}) = \arg\min_{z \in Y} \sum_{y \in Y} \lambda_{yz} p(y) p(\mathbf{x}|y)$$

- **Corollary**: If $\lambda_{yy} = 0$ and $\lambda_{yz} = \lambda_y$ for all $y, z \in Y$, then

$$h_{\text{opt}}(\mathbf{x}) = \arg\max_{y \in Y} \lambda_y p(\mathbf{x}|y) p(y)$$



Bayesian Classification

- **Theoretical setup**:
  — Assumption: we know probabilities $p(y)$ and $p(\mathbf{x}|y)$, $y \in Y$
  — We now how to construct a classifier $h(\mathbf{x})$, minimizing average risk $R(h)$
- **Applied setup**:
  — Assumption: training set $S_m = \{(\mathbf{x}_i, y_i)\}_{i=1}^m$
  — Estimate probabilities $\hat{p}(y)$ and $\hat{p}(\mathbf{x}|y)$, $y \in Y$ to calculate a Bayesian classifier
- We loose optimality when using empirical probability estimates
- Usually it is more difficult to estimate probability density function then to construct efficient classifier

Maximum Likelihood Estimate

- **Assumption**: parametric class of probability density functions
$$p(\mathbf{x}) \in \{f(x; \boldsymbol{\theta}),\, \boldsymbol{\theta} \in \Theta\},$$
where $\boldsymbol{\theta}$ are some parameters. We assume that there exists some $\boldsymbol{\theta}^*$ s.t. $p(\mathbf{x}) = f(x; \boldsymbol{\theta}^*)$
- **Problem**: using i.i.d. sample $X_m = \{\mathbf{x}_1, \ldots, \mathbf{x}_m\}$ estimate $\boldsymbol{\theta}$
- **Log-Likelihood function**:
$$L(\boldsymbol{\theta}; X_m) = \log \prod_{i=1}^{m} f(\mathbf{x}_i; \boldsymbol{\theta}) = \sum_{i=1}^{m} \log f(\mathbf{x}_i; \boldsymbol{\theta})$$
- **Maximum Likelihood Estimate**:
$$\hat{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta}} L(\boldsymbol{\theta}; X_m)$$

Kullback-Leibler Divergence

- Kullback-Leibler divergence is equal to
$$D_{KL}(f|g) = \int_X f(\mathbf{x}) \log\left(\frac{f(\mathbf{x})}{g(\mathbf{x})}\right) d\mathbf{x}$$
- We can prove that $D_{KL}(f|g) \geq 0$, $D_{KL}(g|g) = 0$ and $D_{KL}(\cdot|\cdot)$ is not symmetric
- Let us denote by
$$KL_m(\theta) = \frac{1}{m}\sum_{i=1}^{m} \log\frac{f(\mathbf{x}_i;\boldsymbol{\theta})}{f(\mathbf{x}_i;\boldsymbol{\theta}^*)},$$
then $KL_m(\theta) = \frac{1}{m}(L(\boldsymbol{\theta};X_m) - L(\boldsymbol{\theta}^*;X_m))$
- Thanks to the Law of Large Numbers a.s. $KL_m(\theta) \to D_{KL}(f(\cdot;\boldsymbol{\theta}^*)|f(\cdot;\boldsymbol{\theta}))$ when $m \to \infty$. Thus, maximization of log-likelihood is equivalent to minimization of KL divergence

Multidimensional Gaussian Distribution

- We assume that $X = \mathbb{R}^N$ and

$$p(\mathbf{x}|y) = \mathcal{N}(\mathbf{x}; \mu_y, \Sigma_y) = \frac{1}{\sqrt{(2\pi)^N \det \Sigma_y}} e^{-\frac{1}{2}(\mathbf{x}-\mu_y)^{\mathrm{T}} \Sigma_y^{-1} (x-\mu_y)},$$

where $\mu_y \in \mathbb{R}^N$, $\Sigma_y \in \mathbb{R}^{N \times N}$, $y \in Y$

- Optimal Separating Boundary

$$B = \left\{ x \in X : \lambda_y p(y) p(\mathbf{x}|y) = \lambda_z p(z) p(\mathbf{x}|z) \right\},$$

where $y, z \in Y$, $y \neq z$

Quadratic Discriminant Function I

- In a general case we get a Quadratic Discriminant function

$$h(\mathbf{x}) = \arg\max_{y \in Y} \Big(\log(\lambda_y p(y)) - \frac{1}{2}(\mathbf{x} - \mu_y)^{\mathrm{T}} \Sigma_y^{-1} (\mathbf{x} - \mu_y) \\ - \frac{1}{2} \log \det \Sigma_y \Big)$$

- MLE for μ_y and Σ_y are

$$\hat{\mu}_y = \frac{1}{m_y} \sum_{i: y_i = y} \mathbf{x}_i, \; \hat{\Sigma}_y = \frac{1}{m_y} \sum_{i: y_i = y} (\mathbf{x}_i - \hat{\mu}_y)(\mathbf{x}_i - \hat{\mu}_y)^{\mathrm{T}}$$

Discriminant Analysis: comments

- Regularization of a covariance matrix estimate
  — Use $\hat{\Sigma} + \tau \mathrm{I}$ instead of $\hat{\Sigma}$
  — Select τ using cross-validation
- Before constructing a discriminator perform outlier detection and censoring of data

Quadratic Discriminant Function II

- E.g. when $Y = \{0, 1\}$ we get the decision boundary

$$
\begin{aligned}
B =& \Big\{x \in X : \log \frac{p(1)\lambda_1}{p(0)\lambda_0} - \frac{1}{2} \log \frac{\det(\Sigma_1)}{\det(\Sigma_0)} \\
& + \mathbf{x}^{\mathrm{T}} \left(\Sigma_1^{-1}\mu_1 - \Sigma_0^{-1}\mu_0\right) - \frac{1}{2}\mathbf{x}^{\mathrm{T}} \left(\Sigma_1^{-1} - \Sigma_0^{-1}\right) \mathbf{x} \\
& - \frac{1}{2} \left(\mu_1^{\mathrm{T}}\Sigma_1^{-1}\mu_1 - \mu_0^{\mathrm{T}}\Sigma_0^{-1}\mu_0\right) = 0\Big\}
\end{aligned}
$$

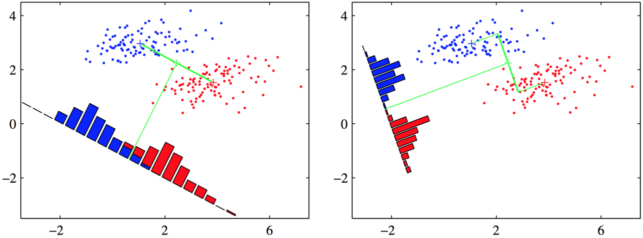
Linear Discriminant Function

- In case when $\Sigma_y = \Sigma$ for all $y \in Y$ we get a Linear Discriminant function
- E.g. when $Y = \{0, 1\}$ we get the decision boundary

$$B = \Big\{x \in X : [\Sigma^{-1}(\mu_1 - \mu_0)]^{\mathrm{T}}\mathbf{x} + \frac{1}{2}\mu_1^{\mathrm{T}}\Sigma^{-1}\mu_1 - \frac{1}{2}\mu_2^{\mathrm{T}}\Sigma^{-1}\mu_2 + \log\frac{p(1)}{p(0)} = 0\Big\}$$

FISHER'S LINEAR DISCRIMINANT I

- We consider a two-class classification problem, i.e. $Y = \{0, 1\}$
- We consider a separating hyperplane
$$h(\mathbf{x}) = \mathbf{w}^{\mathrm{T}}\mathbf{x}$$
and we classify $\mathbf{x} : h(\mathbf{x}) \geq -w_0$ for some w_0 as class C_0 and otherwise as class C_1
- We want to select such projection direction that maximizes the class separation



Fisher's Linear Discriminant II

- Let us denote by m_0 and m_1 the number of points, belonging to classes C_1 and C_2 correspondingly
- We define mean vectors of the two classes as

$$\mathbf{m}_0 = \frac{1}{m_0} \sum_{i:y_i=0} \mathbf{x}_i,\ \mathbf{m}_1 = \frac{1}{m_1} \sum_{i:y_i=1} \mathbf{x}_i$$

- The simplest measure of separation = separation of the projected class means, i.e.

$$m_{1,\mathbf{w}} - m_{0,\mathbf{w}} = \mathbf{w}^{\mathrm{T}}(\mathbf{m}_1 - \mathbf{m}_0)$$

- The within-class variance of the transformed data from class C_k is given by

$$s_k^2 = \sum_{i:y_i=k} (z_i - m_{k,\mathbf{w}})^2,\ k \in \{0, 1\}$$

where $z_i = \mathbf{w}^{\mathrm{T}}\mathbf{x}_i$

Fisher's Linear Discriminant II

- The Fisher criterion is
$$J(\mathbf{w}) = \frac{(m_{1,\mathbf{w}} - m_{0,\mathbf{w}})^2}{s_1^2 + s_0^2}$$
- In vector form
$$J(\mathbf{w}) = \frac{\mathbf{w}\mathbf{S}_B\mathbf{w}}{\mathbf{w}\mathbf{S}_W\mathbf{w}},$$
where
$$\mathbf{S}_B = (\mathbf{m}_1 - \mathbf{m}_0)(\mathbf{m}_1 - \mathbf{m}_0)^{\mathrm{T}}$$
is the between-class covariance matrix, and
$$\mathbf{S}_W = \sum_{i:y_i=0}(\mathbf{x}_i - \mathbf{m}_0)(\mathbf{x}_i - \mathbf{m}_0)^{\mathrm{T}} + \sum_{i:y_i=1}(\mathbf{x}_i - \mathbf{m}_1)(\mathbf{x}_i - \mathbf{m}_1)^{\mathrm{T}}$$
is the within-class covariance matrix
- $J(\mathbf{w})$ is maximized for
$$\mathbf{w} \sim \mathbf{S}_W^{-1}(\mathbf{m}_1 - \mathbf{m}_0)$$

Fisher's discriminant for multiple classes

- We consider $K > 2$ classes ($N > K$)
- The weight vectors $\{\mathbf{w}_k\}$ can be consider as columns of a matrix $\mathbf{W}$, s.t.
$$\mathbf{y} = \mathbf{W}\mathbf{x},$$
i.e. $h_k(\mathbf{x}) = \mathbf{w}_k^{\mathrm{T}}\mathbf{x}$ and we assign a point $\mathbf{x}$ to class C_k if $h_k(\mathbf{x}) > h_j(\mathbf{x})$ for all $j \neq k$
- The generalization of the within-class covariance is $\mathbf{S}_W = \sum_{k=1}^{K} \mathbf{S}_k$, where for $k \in \{0, 1, \dots, K-1\}$

$$\mathbf{S}_k = \sum_{i:y_i=k} (\mathbf{x}_i - \mathbf{m}_k)(\mathbf{x}_i - \mathbf{m}_k)^{\mathrm{T}},\ \mathbf{m}_k = \frac{1}{m_k} \sum_{i:y_i=k} \mathbf{x}_i,$$

Fisher's discriminant for multiple classes II

- The total covariance matrix

$$\mathbf{S}_T = \sum_{i=1}^{m} (\mathbf{x}_i - \mathbf{m})(\mathbf{x}_i - \mathbf{m})^{\mathrm{T}},$$

where the mean of the total data set $\mathbf{m} = \frac{1}{m}\sum_{i=1}^{m} \mathbf{x}_i$, and

$$\mathbf{S}_T = \mathbf{S}_W + \mathbf{S}_B,\ \mathbf{S}_B = \sum_{k=0}^{K-1} m_k(\mathbf{m}_k - \mathbf{m})(\mathbf{m}_k - \mathbf{m})^{\mathrm{T}}$$

- Let us consider *projected* features $\mathbf{z} = \mathbf{W}\mathbf{x}$. In this space we can analogously defined matrices

$$\mathbf{s}_W = \sum_{k=0}^{K-1} \sum_{i:y_i=k} (\mathbf{z}_i - \mu_k)(\mathbf{z}_i - \mu_k)^{\mathrm{T}},\ \mathbf{s}_B = \sum_{k=0}^{K-1} m_k(\mu_k - \mu)(\mu_k - \mu)^{\mathrm{T}},$$

where $\mu_k = \frac{1}{m_k}\sum_{i:y_i=k} \mathbf{z}_i$, $\mu = \frac{1}{m}\sum_{k=0}^{K-1} m_k \mu_k$

Fisher's discriminant for multiple classes III

- We want to construct a scalar that is large when the between-class covariance is large, and when the within-class covariance is small
- One example is given by

$$J(\mathbf{W}) = \mathrm{Tr}\left\{\mathbf{s}_W^{-1}\mathbf{s}_B\right\} = \mathrm{Tr}\left\{(\mathbf{W}\mathbf{S}_W\mathbf{W}^{\mathrm{T}})^{-1}(\mathbf{W}\mathbf{S}_B\mathbf{W})^{\mathrm{T}}\right\}$$

- The weight vectors are determined by those eigenvectors of $\mathbf{S}_W^{-1}\mathbf{S}_B$ that correspond to the largest eigenvalues

1. Optimal Bayesian Classifier
2. Discriminant Analysis
3. Learning a Classifier
4. Logistic Regression

Notations

- Learning sample $S_m = \{(\mathbf{x}_i, y_i)\}_{i=1}^m$, $\mathbf{x}_i \in \mathbb{R}^N$, $y_i \in \{-1, +1\}$
- Linear Classifier

$$h(\mathbf{x}; \mathbf{w}) = \mathrm{sign}(\mathbf{w}^{\mathrm{T}} \mathbf{x})$$

- Binary Loss function and its (upper bound) approximation

$$1_{(\mathbf{w}^{\mathrm{T}} \mathbf{x}_i) y_i < 0} \leq L((\mathbf{w}^{\mathrm{T}} \mathbf{x}_i) y_i)$$

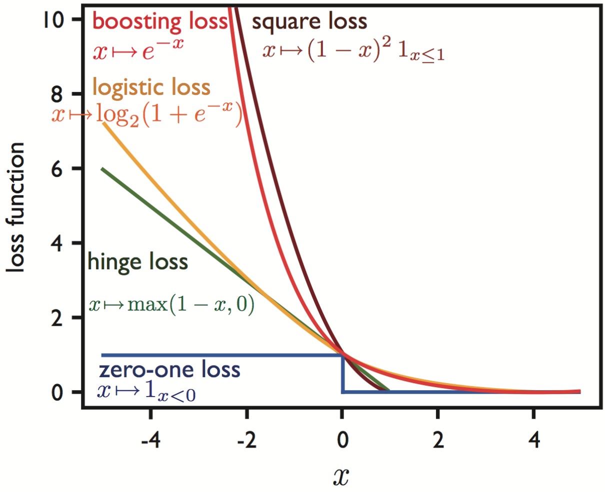
- Learning $\equiv$ ERM

$$R(\mathbf{w}) = \sum_{i=1}^m 1_{(\mathbf{w}^{\mathrm{T}} \mathbf{x}_i) y_i < 0} \leq \sum_{i=1}^m L((\mathbf{w}^{\mathrm{T}} \mathbf{x}_i) y_i) \to \min_{\mathbf{w}}$$

- Testing using a separate sample $\tilde{S}_{\tilde{m}} = \{(\tilde{x}_j, \tilde{y}_j)\}_{j=1}^{\tilde{m}}$

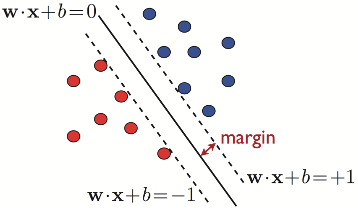
$$\tilde{R}(\mathbf{w}) = \sum_{i=1}^{\tilde{m}} 1_{(\mathbf{w}^{\mathrm{T}} \tilde{x}_i) \tilde{y}_i < 0}$$

Surrogate Loss Functions



Margin: general case

- Binary classification $y_i \in \{-1, +1\}$, binary classifier
$$h(\mathbf{x}; \mathbf{w}) = \mathrm{sign}[g(\mathbf{w}, \mathbf{x})]$$
- Here
  - — $g(\mathbf{w}, \mathbf{x})$ is a separating (discriminating) function
  - — $g(\mathbf{w}, \mathbf{x}) = 0$ is an equation of a separating surface
- **Definition**: $\rho(\mathbf{x}_i; \mathbf{w}) = g(\mathbf{w}, \mathbf{x}_i) y_i$ is a margin of an object $\mathbf{x}_i$, i.e. if $\rho(\mathbf{x}_i, \mathbf{w}) < 0$ then this is an error
- In a linear case for $\|\mathbf{w}\| = 1$ the geometric margin is equal to $\rho(\mathbf{x}_i; \mathbf{w}) = (\mathbf{w}^{\mathrm{T}} \mathbf{x}_i) y_i$



Overfitting

- Causes of overfitting
  — too small number of examples
  — too big number of features
  — linear dependence between features (multicollinearity)
- Symptoms of Overfitting
  — too big absolute values of weights $|w_j|$ and different signs of w_j
  — $R(\omega) \ll \tilde{R}(\omega)$ (test error is $\gg$ than train error)
- Regularization is typically used to prevent overfitting

Regularization

- We impose additional penalty for high absolute values of weights

$$\overline{L}(\mathbf{w}; y) = \sum_{i=1}^{m} L((\mathbf{w}^{\mathrm{T}}\mathbf{x}_i)y_i) + \frac{\lambda}{2}\|\mathbf{w}\|^2$$

- In order to tune regularization coefficient λ we can use
  — cross-validation
  — Bayesian inference

Maximum Likelihood

- Let $p(\mathbf{x}, y|\mathbf{w}) = p(y|\mathbf{x}, \mathbf{w})p(\mathbf{x})$ be some probability distribution on $X \times Y$
- MLE for $\mathbf{w}$

$$\prod_{i=1}^{m} p(\mathbf{x}_i, y_i|\mathbf{w}) = \prod_{i=1}^{m} p(y_i|\mathbf{x}_i, \mathbf{w})p(\mathbf{x}_i) \to \max_{\mathbf{w}}$$

- Log-likelihood

$$\mathcal{L}(\mathbf{w}) = \sum_{i=1}^{m} \log p(y_i|\mathbf{x}_i, \mathbf{w}) \to \max_{\mathbf{w}}$$

- For two classes $Y = \{0, 1\}$ and $h(\mathbf{x}, \mathbf{w}) = p(y = 1|\mathbf{x}, \mathbf{w})$ we get that

$$\sum_{i=1}^{m} y_i \log h(\mathbf{x}_i, \mathbf{w}) + \sum_{i=1}^{m} (1 - y_i) \log(1 - h(\mathbf{x}_i, \mathbf{w})) \to \max_{\mathbf{w}}$$

MLE vs. ERM

- MLE

$$\mathcal{L}(\mathbf{w}) = \sum_{i=1}^{m} \log p(y_i|\mathbf{x}_i, \mathbf{w}) \to \max_{\mathbf{w}}$$

- Minimization of Approximated Empirical Risk

$$L(\mathbf{w}) = \sum_{i=1}^{m} L\left(y_i g(\mathbf{x}_i, \mathbf{w})\right) \to \min_{\mathbf{w}}$$

- If we set

$$-\log p(y_i|\mathbf{x}_i, \mathbf{w}) = L\left(y_i g(\mathbf{x}_i, \mathbf{w})\right),$$

we will get the same results, i.e. the surrogate loss function $L(\cdot)$ and $g(\mathbf{x}, \mathbf{w})$ define the model $p(y|\mathbf{x}, \mathbf{w})$

Two-Class Logistic Regression

- Linear Classifier in case of $Y = \{-1, +1\}$

$$h(\mathbf{x}) = \mathrm{sign}(\mathbf{w}^{\mathrm{T}}\mathbf{x})$$

- Margin is equal to $\rho = (\mathbf{w}^{\mathrm{T}}\mathbf{x})y$
- Logarithmic loss function

$$L(t) = \log(1 + e^{-t})$$

- A model for conditional probability

$$p(y|\mathbf{x}, \mathbf{w}) = \sigma(\mathbf{w}^{\mathrm{T}}\mathbf{x}) = \frac{1}{1 + e^{-\mathbf{w}^{\mathrm{T}}\mathbf{x}}}$$

- Regularized logistic regression

$$\overline{L}(\mathbf{w}) = \sum_{i=1}^{m} \log(1 + \exp(-(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i)y_i)) + \frac{\lambda}{2}\|\mathbf{w}\|^2 \to \min_{\mathbf{w}}$$

MultiClass Logistic Regression

- Linear Classifier in a multiclass case, i.e. $\#(Y) > 1$
- Probability of an object to belong to some class y is equal to

$$p(y|\mathbf{x}, \mathbf{w}) = \frac{\exp(\mathbf{w}_y^{\mathrm{T}}\mathbf{x})}{\sum_{z \in Y} \exp(\mathbf{w}_z^{\mathrm{T}}\mathbf{x})} = \mathrm{SoftMax}_{y \in Y}(\mathbf{w}_y^{\mathrm{T}}\mathbf{x})$$

- Regularized logistic regression

$$\overline{L}(\mathbf{w}) = \sum_{i=1}^{m} \log p(y_i|\mathbf{x}_i, \mathbf{w}) + \frac{\lambda}{2} \sum_{y \in Y} \|\mathbf{w}_y\|^2 \to \min_{\mathbf{w}}$$

The Newton-Raphson Method

- Surrogate Loss function for a binary logistic regression

$$R(\mathbf{w}) = \sum_{i=1}^{m} L((\mathbf{w}^{\mathrm{T}}\mathbf{x}_i)y_i)$$

- Steps:

$$\mathbf{w}^{t+1} = \mathbf{w}^{t} - r_t(R''(\mathbf{w}^{t}))^{-1}R'(\mathbf{w}^{t})$$

- Components of a gradient

$$\frac{\partial R(\mathbf{w})}{\partial w_j} = -\sum_{i=1}^{m}(1-\sigma_i)y_i x_{i,j},\ j = 1,\dots,N$$

- Hessian

$$\frac{\partial^2 R(\mathbf{w})}{\partial w_j \partial w_k} = \sum_{i=1}^{m}(1-\sigma_i)\sigma_i x_{i,j} x_{i,k},\ j,k = 1,\dots,N,$$

where $\sigma_i = \sigma(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i y_i)$, $\sigma(t) = \frac{1}{1+e^{-t}}$

Notations

- $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^m \in \mathbb{R}^{m\times N}$ is a matrix of objects features
- $\Gamma = \mathrm{diag}\left(\sqrt{(1-\sigma_i)\sigma_i}\right) \in \mathbb{R}^{m\times m}$ is a diagonal matrix of weights
- $\tilde{\mathbf{X}} = \Gamma\mathbf{X}$ is a weighted matrix of features
- $\tilde{y}_i = y_i\sqrt{(1-\sigma_i)/\sigma_i}$, $\tilde{\mathbf{y}} = \{\tilde{y}_i\}_{i=1}^m$ is a weighted vector of labels
- Then we get that

$$(R''(\mathbf{w}))^{-1}R'(\mathbf{w}) = -\left(\mathbf{X}^{\mathrm{T}}\Gamma^2\mathbf{X}\right)^{-1}\mathbf{X}^{\mathrm{T}}\Gamma\tilde{\mathbf{y}} = -\left(\tilde{\mathbf{X}}^{\mathrm{T}}\tilde{\mathbf{X}}\right)^{-1}\tilde{\mathbf{X}}\tilde{\mathbf{y}}$$

This coincides with a solution of a weighted least-squares problem

$$R(\mathbf{w}) = \left\|\tilde{\mathbf{X}}\mathbf{w} - \tilde{\mathbf{y}}\right\|^2 = \sum_{i=1}^m (1-\sigma_i)\sigma_i\left(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i - \frac{y_i}{\sigma_i}\right)^2 \to \min_{\mathbf{w}}$$

Interpretation

- On each step of the Newton-Raphson method we construct a weighted least-squares regression

$$R(\mathbf{w}) = \sum_{i=1}^{m} (1-\sigma_i)\sigma_i \left(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i - \frac{y_i}{\sigma_i}\right)^2 \to \min_{\mathbf{w}}$$

- Here
  — $\sigma_i = p(y_i|\mathbf{x}_i)$ is a probability to correctly classify $\mathbf{x}_i$
  — the close $\mathbf{x}_i$ to the boundary, the bigger the weight $(1-\sigma_i)\sigma_i$
  — the bigger the probability of an error, the bigger the value of y_i/σ_i

  Thus on each iteration we tune $\mathbf{w}$ to perform better on more difficult examples

Iterativey Reweighted Least Squares

- **Input**: $\mathbf{X}$, $\mathbf{y}$, i.e. a matrix and a vector of input features and corresponding labels
- **Output**: estimate of $\mathbf{w}$
- For $t = 1, 2, \dots$
  - — $\sigma_i = \sigma(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i y_i)$, $i = 1, \dots, m$
  - — $\gamma_i = \sqrt{(1-\sigma_i)\sigma_i}$, $i = 1, \dots, m$
  - — $\tilde{\mathbf{X}} = \mathrm{diag}(\gamma_1, \dots, \gamma_m)\mathbf{X}$
  - — $\tilde{y}_i = y_i\sqrt{(1-\sigma_i)/\sigma_i}$, $i = 1, \dots, m$
  - — select a gradient step r_t and calculate $\mathbf{w} \leftarrow \mathbf{w} + r_t \left(\tilde{F}^{\mathrm{T}}\tilde{F}\right)^{-1} \tilde{F}^{\mathrm{T}}\mathbf{y}$
  - — if σ_i changes not significantly, then stop iterations