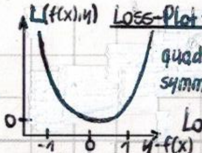
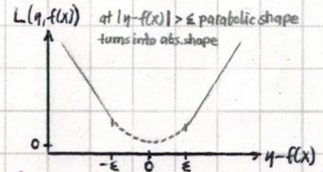
Supervised Regression



Loss-Plot for L2 loss
quadratic loss curve
symmetric

Loss characterized as function of residuals $r = y - f(x)$



Design matrix

Let $\underline{x}_1, \dots, \underline{x}_p$ be our p feature vectors, then we call the matrix
$X = \begin{pmatrix} 1 & | & | & & | \\ \vdots & \underline{x}_1 & \underline{x}_2 & \cdots & \underline{x}_p \\ 1 & | & | & & | \end{pmatrix} \in \mathbb{R}^{n \times (p+1)}$ the design matrix.
Furthermore, the define $\underline{x} := (1, x_1, \dots, x_p)^T \in \mathbb{R}^{p+1}$ and $\underline{\theta} = (\theta_0, \theta_1, \dots, \theta_p) \in \mathbb{R}^{p+1}$

one row as col. vector (X has n of those rows), s.nt. $\tilde{X} = (1, X)$ for clarity to distinguish intercept / feature vec.

Setup

We predict $y \in \mathbb{R}$ as linear combinations of features

$\hat{y} = f(\underline{x}) = \underline{\theta}^T \underline{x} = \theta_0 + \theta_1 x_1 + \dots + \theta_p x_p$

This results in the hypothesis space $\mathcal{H} = \{ f(\underline{x}) = \underline{\theta}^T \underline{x} \mid \underline{\theta} \in \mathbb{R}^{p+1} \}$

A typical choice for the Loss-function is L2-Loss.

This results in $R_{emp}(\underline{\theta}) = \sum_{i=1}^{n} (y^{(i)} - \underline{\theta}^T \underline{x}^{(i)})^2 = \text{SSE} : \sum_{i=1}^{n} (\underline{\varepsilon})^2 \quad \text{MSE} : \frac{1}{n} \cdot \text{SSE}$

L2 - Loss / Squared Error

penalizes quadr. residuals

$L(y, f(\underline{x})) = \frac{1}{2}(y - f(\underline{x}))^2$ or $L(y, f(\underline{x})) = (y - f(\underline{x}))^2$

Error $\perp$ Feature Space $X^T(y - X\theta) = 0$ easier to optimize

Properties: Convex; differentiable everywhere

Problem: Outliers have a huge effect on L2.

L1 - Loss / Absolute Error

penalizes absolute residuals

$L(y, f(\underline{x})) = |y - f(\underline{x})|$

Properties: Convex; Robust against outliers

Problem: Not differentiable, slower to optimize (only numerically)

Huber Loss

Weighted piecewise combo of L1, L2 loss

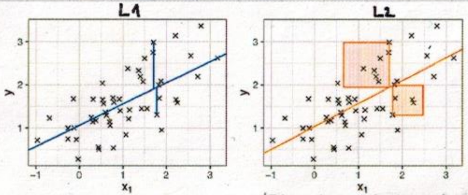
ε marks where L2 transits to L1 loss ($\varepsilon > 0$)

$$L(y, f(x)) = \begin{cases} \frac{1}{2}(y - f(x))^2 & \text{if } |y - f(x)| \leq \varepsilon \\ \varepsilon |y - f(x)| - \frac{1}{2}\varepsilon^2 & \text{else} \end{cases}$$

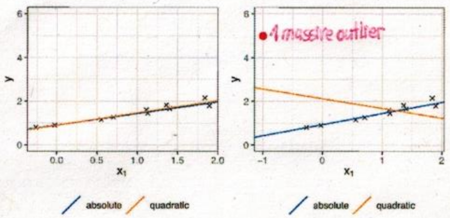
Huber loss combines advantages of both

Its smooth & differentiable like L2 loss

Doesn't punish large residuals harshly just as L1 loss => more robust



L2 loss: Connect to classical LM assumption of gaussian error distr.



Optimization

We want to find $\hat{\underline{\theta}} = \arg\min_{\underline{\theta} \in \Theta} \sum_{i=1}^{n} (y^{(i)} - \underline{\theta}^T \underline{x}^{(i)})^2 = \arg\min_{\underline{\theta} \in \Theta} \| \underline{y} - X\underline{\theta} \|_2^2$

This results in the ordinary-least-squares estimator:

$\hat{\underline{\theta}} = (X^T X)^{-1} X^T \underline{y}$ (MLE)

Gradient: $\nabla_\theta R_{emp}(\theta) = \frac{\partial}{\partial \theta} \|y - X\theta\|_2^2 = (y - X\theta)^T (y - X\theta)$
$= -2y^T X + 2\theta^T X^T X = 2(-y^T X + \theta^T X^T X)$

ANALYTICAL OPTIMIZATION – PROOF

$\mathcal{R}_{emp}(\theta) = \sum_{i=1}^{n} (\underbrace{y^{(i)} - \theta^\top \mathbf{x}^{(i)}}_{=:\epsilon_i})^2 = \| \underbrace{\mathbf{y} - \mathbf{X}\theta}_{=:\epsilon} \|_2^2; \quad \theta \in \mathbb{R}^{\tilde{p}}$ with $\tilde{p} := p + 1$

using L2 loss

(sum notation)

$$\begin{aligned}
0 &= \frac{\partial \mathcal{R}_{emp}(\theta)}{\partial \theta} \\
0 &= \frac{\partial}{\partial \theta} \sum_{i=1}^{n} \epsilon_i^2 \quad \Big| \text{ sum \& chain rule} \\
0 &= \sum_{i=1}^{n} \frac{\partial \epsilon_i^2}{\partial \epsilon_i} \frac{\partial \epsilon_i}{\partial \theta} \\
0 &= \sum_{i=1}^{n} 2\epsilon_i (-1) (\mathbf{x}^{(i)})^\top \\
0 &= \sum_{i=1}^{n} (y^{(i)} - \theta^\top \mathbf{x}^{(i)}) (\mathbf{x}^{(i)})^\top \\
\sum_{i=1}^{n} \theta^\top \mathbf{x}^{(i)} (\mathbf{x}^{(i)})^\top &= \sum_{i=1}^{n} y^{(i)} (\mathbf{x}^{(i)})^\top \quad \Big| \text{ transpose} \\
\theta \sum_{i=1}^{n} \underbrace{(\mathbf{x}^{(i)})^\top \mathbf{x}^{(i)}}_{1 \times 1} &= \sum_{i=1}^{n} \mathbf{x}^{(i)} y^{(i)} \\
\theta &= \underbrace{\sum_{i=1}^{n} (\underbrace{(\mathbf{x}^{(i)})^\top \mathbf{x}^{(i)}}_{1 \times 1})^{-1} \underbrace{\mathbf{x}^{(i)}}_{\tilde{p} \times 1} \underbrace{y^{(i)}}_{1 \times 1}}_{\tilde{p} \times 1}
\end{aligned}$$

(matrix notation)

$$\begin{aligned}
0 &= \frac{\partial \mathcal{R}_{emp}(\theta)}{\partial \theta} \\
0 &= \frac{\partial \|\epsilon\|_2^2}{\partial \theta} \\
0 &= \frac{\partial \epsilon^\top \epsilon}{\partial \theta} \quad \Big| \text{ chain rule} \\
0 &= \frac{\partial \epsilon^\top \epsilon}{\partial \epsilon} \cdot \frac{\partial \epsilon}{\partial \theta} \\
0 &= 2\epsilon^\top \cdot (-1 \cdot \mathbf{X}) \\
0 &= (\mathbf{y} - \mathbf{X}\theta)^\top \mathbf{X} \\
0 &= \mathbf{y}^\top \mathbf{X} - \theta^\top \mathbf{X}^\top \mathbf{X} \\
\theta^\top \mathbf{X}^\top \mathbf{X} &= \mathbf{y}^\top \mathbf{X} \quad \Big| \text{ transpose} \\
\mathbf{X}^\top \mathbf{X} \theta &= \mathbf{X}^\top \mathbf{y} \\
\theta &= \underbrace{\underbrace{(\mathbf{X}^\top \mathbf{X})^{-1}}_{\tilde{p} \times \tilde{p}} \underbrace{\mathbf{X}^\top}_{\tilde{p} \times n} \underbrace{\mathbf{y}}_{n \times 1}}_{\tilde{p} \times 1}
\end{aligned}$$

Ref. coding is basically One-Hot-Enc. with one Ref. categ. permanent disabled $= 0$ (for categorical features)

Sidenote: Linear Regression with intercept must use *leave one out* dummy encoding!

One-Hot-Encoding (full k levels for k categories) => one redundant level. Leads to "over-parameterization" and thus an indistinguishable non-unique parameter vector, unstable possibly ridiculous θ estimates

$X_1 = 1 - X_2 - X_3 - \dots X_p$ (All $= 0$, only "hot" $X_1 = 1$ => $(\beta_0, \beta_1, \dots \beta_p)$ e.g. using $X_1 = 1 - X_2$ $= (\beta_0 + \beta_1, 0, \beta_2 - \beta_1, \beta_3, \dots, \beta_p)$

Analytically: X (=> $X^T X$ also) not invertible due to lineary dependancy of encoded categ. features
perfect multicollinearity / dummy variable trap

But: Regularized Regression requires full encoding, otherwise intercept not penalized!

Polynomial Regression

General Linear Model

We predict $y \in \mathbb{R}$ as linear combinations of basis functions ϕ_j

$\hat{y} = f(\underline{x}) = \theta_0 + \theta_1 \phi_1(x_1) + \dots + \theta_p \phi_p(x_p)$

If we set $\phi_j = id_x$ we get the usual linear regression model.

! This model is linear in parameters θ but not in covariates / features $\underline{x}$.

$f(a\underline{x} + b\underline{x}^* ; \underline{\theta}) \neq a f(\underline{x}; \underline{\theta}) + b f(\underline{x}^*; \underline{\theta})$ but $f(\underline{x}; a\underline{\theta} + b\underline{\theta}^*) = a f(\underline{x}; \underline{\theta}) + b f(\underline{x}; \underline{\theta}^*)$

The design matrix now is defined as $X = \begin{pmatrix} 1 & | & & | \\ \vdots & \phi_1(\underline{x}_1) & \dots & \phi_p(\underline{x}_p) \\ 1 & | & & | \end{pmatrix}$ can still optimize with normal equation

Overfitting risk with complex polynomials / basis functions

How high can/should we go with degree of modelling polynomials? (Hyperparameter!)

Problem with high degree: Our model will fit the training data excellent, but generally will not perform well on new (test) data

- Data contains random noise that is not part of true data-generating process
  => model with overly high capacity learns all those spurious patterns and those will not reappear in new data, leading to poor generalization of our model
- Also higher degrees can lead to instability (oscillation esp. at bounds, "Runges Phenomenon")

Polynomial Regression

More flexible than simple linear regression

In the Polynomial Regression we choose the basis functions $\phi^{(d)} : \mathbb{R} \to \mathbb{R}, \; x_j \mapsto \sum_{k=1}^{d} \beta_k x_j^k$

This results in $f(\underline{x}) = \theta_0 + \theta_1 \phi^{(d)}(\underline{x}) = \theta_0 + \sum_{k=1}^{d} \theta_{1,k} x^k$ where $\theta_{1,k} := \theta_1 \cdot \beta_k$ (θ_1 is just param. for full $\phi^{(d)}(x)$)

here just one feature X

idea: merge θ_1 and weight β_k (still linear in merged parameter)

$$f(\mathbf{x}^{(i)}) = \sum_{j=1}^{p} \sum_{k=0}^{d} \theta_{j,k} (x_j^{(i)})^k$$

$X = \begin{pmatrix} 1 & x^{(1)} & (x^{(1)})^2 & \dots & (x^{(1)})^d \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & x^{(n)} & (x^{(n)})^2 & \dots & (x^{(n)})^d \end{pmatrix}, \; \underline{\theta} \in \mathbb{R}^{d+1}$

For two features we might model like this:

$f(x) = \theta_0 + \theta_1 x_1 + \sum_{k=1}^{d} \theta_{2,k} x_2^k$

linear term x_1 — polynomials for x_2 up until grade d