Regression

Victor Kitov

v.v.kitov@yandex.ru

Yandex
School of Data Analysis.

Table of Contents

1. Linear regression
2. Nonlinear dependencies
3. Regularization & restrictions.
4. Different loss-functions
5. Weighted account for observations
6. Nadaraya-Watson regression
7. Other types of regression

Linear regression

- Linear model $f(x,\beta) = x^T\beta = \sum_{i=1}^{D}\beta_i x^i$
  - we include constant feature in x
- Define $X \in \mathbb{R}^{NxD}$, $\{X\}_{ij}$ defines the j-th feature of i-th object, $Y \in \mathbb{R}^n$, $\{Y\}_i$ - target value for i-th object.
- Ordinary least squares (OLS) method:

$$\sum_{n=1}^{N}\left(x_n^T\beta - y_n\right)^2 \to \min_{\beta}$$

Solution

Stationarity condition:

$$2\sum_{n=1}^{N} x_n \left(x_n^T \beta - y_n\right) = 0$$

In matrix form:

$$2X^T(X\beta - Y) = 0$$

so

$$\widehat{\beta} = (X^T X)^{-1} X^T Y$$

Comments

- This is the global minimum, because the optimized criteria is convex.
- Geometric interpretation:
  - find linear combination of feature measurements that best reproduce Y
  - solution - combinaton of features, giving projection of Y on linear span of feature measurements.

Linearly dependent features

- Solution $\widehat{\beta} = (X^T X)^{-1} X^T Y$ exists when $X^T X$ is non-degenerate
- Problem occurs when one of the features is a linear combination of the other
    - because of the property $\forall X: \; rank(X) = rank(X^T X)$
    - example: constant unity feature c and one-hot-encoding $e_1, e_2, ...e_K$, because $\sum_k e_k \equiv c$
    - interpretation: non-identifiability of $\widehat{\beta}$ for linearly dependent features:
        - linear dependence: $\exists \alpha: \; x^T \alpha = 0 \, \forall x$
        - suppose β solves linear regression $y = x^T \beta$
        - then $x^T \beta \equiv x^T \beta + k x^T \alpha \equiv x^T (\beta + k\alpha)$, so $\beta + k\alpha$ is also a solution!

Linearly dependent features

- Problem may be solved by:
  - feature selection
  - dimensionality reduction
  - imposing additional requirements on the solution (regularization)

Analysis of linear regression

Advantages:

- single optimum, which is global (for non-singular matrix)
- analytical solution
- interpretable solution and algorithm

Drawbacks:

- too simple model assumptions (may not be satisfied)
- $X^T X$ should be non-degenerate (and well-conditioned)

Table of Contents

1. Linear regression
2. Nonlinear dependencies
3. Regularization & restrictions.
4. Different loss-functions
5. Weighted account for observations
6. Nadaraya-Watson regression
7. Other types of regression

Generalization by nonlinear transformations

Nonlinearity by x in linear regression may be achieved by applying non-linear transformations to the features:

$$x \rightarrow [\phi_1(x), \phi_2(x), \dots \phi_M(x)]$$

$$f(x) = \phi(x)^T \beta = \sum_{m=1}^{M} \beta_m \phi_m(x)$$

The model remains to be linear in β, so all advantages of linear regression remain:

- interpretability
- closed form solution
- global optimum

Typical transformations

$\phi_k(x)$	**comments**
$\mathbb{I}\left\{x^i \in [a, b]\right\}$	binarization of feature
$\left(x^i\right)\left(x^j\right)$	interaction of features
$\exp\left\{-\gamma \lVert x - z \rVert^2\right\}$	closeness to some reference point z
$\ln x^k$	alignment of distribution with heavy tails
$F(x^k)$	convert to uniform distribution with c.d.f. of x^k

Non-linear regression

- Alternatively we can model $\mathcal{X} \to \mathcal{Y}$ with arbitrary non-linear function $\widehat{y} = f(x|\theta)$

$$L(\theta|X, Y) = \sum_{n=1}^{N} (f(x_n|\theta) - y_n)^2$$

$$\widehat{\theta} = \arg\min_{\theta} L(\theta|X, Y)$$

- No analytical solution for $\widehat{\theta}$ will exist in general
    - need numeric optimization methods.

Table of Contents

1. Linear regression
2. Nonlinear dependencies
3. Regularization & restrictions.
4. Different loss-functions
5. Weighted account for observations
6. Nadaraya-Watson regression
7. Other types of regression

Regularization

- Insert additional requirement for regularizer $R(\beta)$ to be small:

$$\sum_{n=1}^{N}\left(x_n^T\beta - y_n\right)^2 + \lambda R(\beta) \to \min_{\beta}$$

- $\lambda > 0$ - hyperparameter.
- $R(\beta)$ penalizes complexity of models.

$$\begin{array}{ll} R(\beta) = ||\beta||_1 & \text{Lasso regression} \\ R(\beta) = ||\beta||_2^2 & \text{Ridge regression} \end{array}$$

- Not only *accuracy* matters for the solution but also *model simplicity*!
- λ controls complexity of the model:

Regularization

- Insert additional requirement for regularizer $R(\beta)$ to be small:

$$\sum_{n=1}^{N}\left(x_n^T\beta - y_n\right)^2 + \lambda R(\beta) \to \min_{\beta}$$

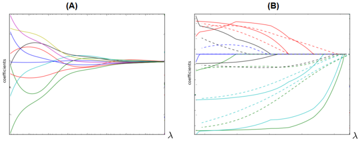
- $\lambda > 0$ - hyperparameter.
- $R(\beta)$ penalizes complexity of models.

$$\begin{array}{ll} R(\beta) = ||\beta||_1 & \text{Lasso regression} \\ R(\beta) = ||\beta||_2^2 & \text{Ridge regression} \end{array}$$

- Not only *accuracy* matters for the solution but also *model simplicity*!
- λ controls complexity of the model:$\uparrow \lambda \Leftrightarrow$ complexity$\downarrow$.

Comments

- Dependency of β from λ for ridge (A) and LASSO (B):



- LASSO can be used for automatic feature selection.
- λ is usually found using cross-validation on exponential grid, e.g. $[10^{-6}, 10^{-5}, ...10^{5}, 10^{6}]$.
- It's always recommended to use regularization because
  - it gives smooth control over model complexity.
  - removes ambiguity for multiple solutions case.

ElasticNet

- ElasticNet:

$$R(\beta) = \alpha||\beta||_1 + (1-\alpha)||\beta||_2^2 \to \min_\beta$$

  $\alpha \in (0,1)$ - hyperparameter, controlling impact of each part.
- If two features x^i and x^j are equal:
  - LASSO may take only one of them
  - ridge will take both with equal weight
    - but it doesn't remove useless features
  - ElasticNet both removes useless features but gives equal weight for usefull equal features
    - good, because feature equality may be due to chance on this particular training set

Ridge regression solution

Ridge regression criterion

$$\sum_{n=1}^{N}\left(x_n^T\beta-y_n\right)^2+\lambda\beta^T\beta\to\min_{\beta}$$

Stationarity condition can be written as:

$$2\sum_{n=1}^{N}x_n\left(x_n^T\beta-y_n\right)+2\lambda\beta=0$$

$$2X^T(X\beta-Y)+\lambda\beta=0$$

$$\left(X^TX+\lambda I\right)\beta=X^TY$$

so

$$\widehat{\beta}=(X^TX+\lambda I)^{-1}X^TY$$

Comments

- $X^TX + \lambda I$ is always non-degenerate as a sum of:
    - non-negative definite X^TX
    - positive definite λI
- Intuition:
    - out of all valid solutions select one giving simplest model
- Other regularizations also restrict the set of solutions.

Different account for different features

- Traditional approach regularizes all features uniformly:

$$\sum_{n=1}^{N}\left(x_n^T\beta-y_n\right)^2+\lambda R(\beta)\to\min_w$$

- Suppose we have K groups of features with indices:

$$I_1,I_2,...I_K$$

- We may control the impact of each group on the model by:

$$\sum_{n=1}^{N}\left(x_n^T\beta-y_n\right)^2+\lambda_1 R(\{\beta_i|i\in I_1\})+...+\lambda_K R(\{\beta_i|i\in I_K\})\to\min_w$$

- $\lambda_1,\lambda_2,...\lambda_K$ can be set using cross-validation
- In practice use common regularizer but with different feature scaling.

Table of Contents

1. Linear regression
2. Nonlinear dependencies
3. Regularization & restrictions.
4. Different loss-functions
5. Weighted account for observations
6. Nadaraya-Watson regression
7. Other types of regression

Idea

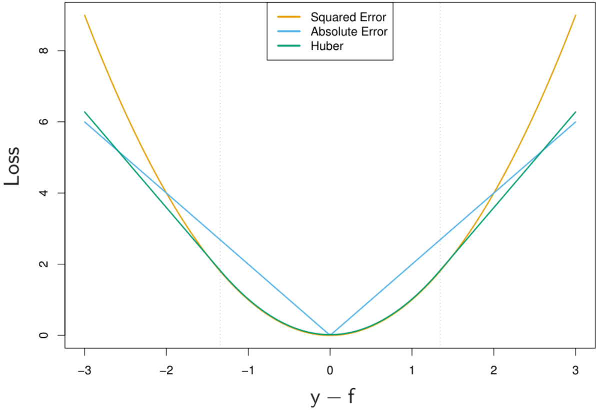
- Generalize quadratic to arbitrary loss:

$$\sum_{n=1}^{N}\left(x^T\beta-y_n\right)^2\to\min_{\beta}\qquad\Longrightarrow\qquad\sum_{n=1}^{N}\mathcal{L}(x_n^T\beta-y_n)\to\min_{\beta}$$

LOSS	NAME	PROPERTIES
$\mathcal{L}(\varepsilon)=\varepsilon^2$	quadratic	differentiable
$\mathcal{L}(\varepsilon)=\lvert\varepsilon\rvert$	absolute	robust
$\mathcal{L}(\varepsilon)=\begin{cases}\frac{1}{2}\varepsilon^2, & \lvert\varepsilon\rvert\le\delta\\ \delta\left(\lvert\varepsilon\rvert-\frac{1}{2}\delta\right) & \lvert\varepsilon\rvert>\delta\end{cases}$	Huber	differentiable, robust

- Robust means solution is robust to outliers in the training set.

Non-quadratic loss functions[1,2]



[1] What is the value of constant prediction, minimizing sum of squared errors?
[2] What is the value of constant prediction, minimizing sum of absolute errors?

Loss function matters

- For $y_1, ...y_N \in \mathbb{R}$ constant minimizers $\widehat{\mu}$:

$$\arg\min_{\mu} \sum_{n=1}^{N} (y_n - \mu)^2 =$$

$$\arg\min_{\mu} \sum_{n=1}^{N} |y_n - \mu| =$$

Loss function matters

- For $y_1, ...y_N \in \mathbb{R}$ constant minimizers $\widehat{\mu}$:

$$\arg\min_{\mu} \sum_{n=1}^{N} (y_n - \mu)^2 = \frac{1}{N} \sum_{n=1}^{N} y_n$$

$$\arg\min_{\mu} \sum_{n=1}^{N} |y_n - \mu| = \text{ median}\{y_1, ...y_N\}$$

Loss function matters

- For $y_1, ...y_N \in \mathbb{R}$ constant minimizers $\widehat{\mu}$:

$$\arg\min_{\mu} \sum_{n=1}^{N} (y_n - \mu)^2 = \frac{1}{N} \sum_{n=1}^{N} y_n$$

$$\arg\min_{\mu} \sum_{n=1}^{N} |y_n - \mu| = \text{median}\{y_1, ...y_N\}$$

- For $x, y \sim P(x, y)$ and functional minimizers $f(x)$:

$$\arg\min_{f(x)} \mathbb{E}\left\{ (f(x) - y)^2 \middle| x \right\} =$$

$$\arg\min_{f(x)} \mathbb{E}\{ |f(x) - y| \,|\, x\} =$$

Loss function matters

- For $y_1, ...y_N \in \mathbb{R}$ constant minimizers $\widehat{\mu}$:

$$\arg\min_{\mu} \sum_{n=1}^{N} (y_n - \mu)^2 = \frac{1}{N} \sum_{n=1}^{N} y_n$$

$$\arg\min_{\mu} \sum_{n=1}^{N} |y_n - \mu| = \text{median}\{y_1, ...y_N\}$$

- For $x, y \sim P(x, y)$ and functional minimizers $f(x)$:

$$\arg\min_{f(x)} \mathbb{E}\left\{ (f(x) - y)^2 \middle| x \right\} = \mathbb{E}[y|x]$$

$$\arg\min_{f(x)} \mathbb{E}\{ |f(x) - y| \,|\, x \} = \text{median}[y|x]$$

Table of Contents

1. Linear regression
2. Nonlinear dependencies
3. Regularization & restrictions.
4. Different loss-functions
5. Weighted account for observations
6. Nadaraya-Watson regression
7. Other types of regression

Weighted account for observations[3]

- Weighted account for observations

$$\sum_{n=1}^{N} w_n (x_n^T \beta - y_n)^2$$

- Weights may be:
    - increased for incorrectly predicted objects
        - algorithm becomes more oriented on error correction
    - decreased for incorrectly predicted objects
        - they may be considered outliers that break our model

[3] Derive solution for weighted regression.

Robust regression

- Initialize $w_1 = ... = w_N = 1/N$
- Repeat:
  - estimate regression $\widehat{y}(x)$ using observations (x_i, y_i) with weights w_i.
  - for each $i = 1, 2, ...N$:
    - re-estimate $\varepsilon_i = \widehat{y}(x_i) - y_i$
    - recalculate $w_i = K\left(|\varepsilon_i|\right)$
  - normalize weights $w_i = \frac{w_i}{\sum_{n=1}^{N} w_n}$

Comments: $K(\cdot)$ is some *decreasing* function, repetition may be

- predefined number of times
- until convergence of model parameters.

Example

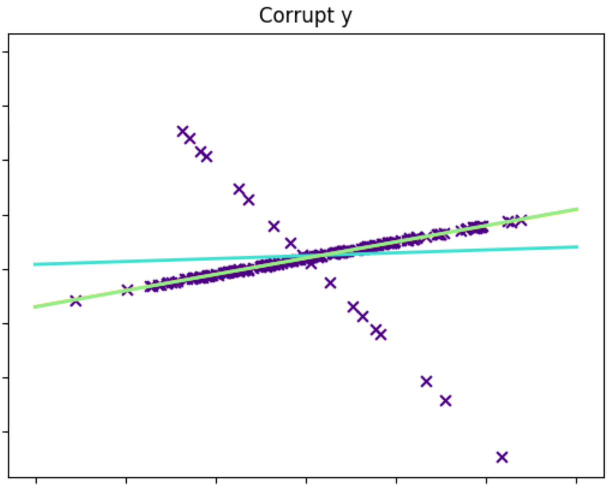


Table of Contents

1. Linear regression
2. Nonlinear dependencies
3. Regularization & restrictions.
4. Different loss-functions
5. Weighted account for observations
6. Nadaraya-Watson regression
7. Other types of regression

Minimum squared error estimate

For training sample $(x_1, y_1), \dots (x_N, y_N)$ consider finding constant $\widehat{y} \in \mathbb{R}$:

$$L(\widehat{y}) = \sum_{i=1}^{N} (\widehat{y} - y_i)^2 \to \min_{\widehat{y} \in \mathbb{R}}$$

$$\frac{\partial L}{\partial \widehat{y}} = 2 \sum_{i=1}^{N} (\widehat{y} - y_i) = 0, \text{ so } \widehat{y} = \frac{1}{N} \sum_{i=1}^{N} y_i$$

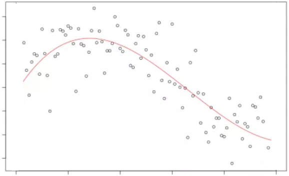
Minimum squared error estimate

For training sample $(x_1, y_1), \ldots (x_N, y_N)$ consider finding constant $\widehat{y} \in \mathbb{R}$:

$$L(\widehat{y}) = \sum_{i=1}^{N} (\widehat{y} - y_i)^2 \to \min_{\widehat{y} \in \mathbb{R}}$$

$$\frac{\partial L}{\partial \widehat{y}} = 2 \sum_{i=1}^{N} (\widehat{y} - y_i) = 0, \text{ so } \widehat{y} = \frac{1}{N} \sum_{i=1}^{N} y_i$$

We need to model general curve $y(x)$:



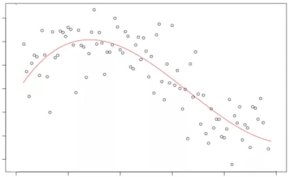
Minimum squared error estimate

For training sample $(x_1, y_1), \dots (x_N, y_N)$ consider finding constant $\widehat{y} \in \mathbb{R}$:

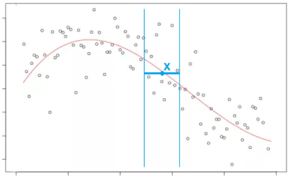
$$L(\widehat{y}) = \sum_{i=1}^{N} (\widehat{y} - y_i)^2 \to \min_{\widehat{y} \in \mathbb{R}}$$

$$\frac{\partial L}{\partial \widehat{y}} = 2 \sum_{i=1}^{N} (\widehat{y} - y_i) = 0, \text{ so } \widehat{y} = \frac{1}{N} \sum_{i=1}^{N} y_i$$

We need to model general curve $y(x)$:



Nadaraya-Watson regression - localized averaging approach.



Nadaraya-Watson regression

- Equivalent names: local constant regression, kernel regression.
- For each x assume $f(x) = const = \alpha$, $\alpha \in \mathbb{R}$.

$$Q(\widehat{y}|x) = \sum_{i=1}^{N} w_i(x)(\widehat{y} - y_i)^2 \to \min_{\alpha \in \mathbb{R}}$$

- Weights depend on the proximity of training objects to the predicted object:

$$w_i(x) = K\left(\frac{\rho(x, x_i)}{h}\right)$$

- $K(u)$ - some decreasing function, called kernel.
- $h(x)$ - some ≥ 0 function called bandwidth.
    - Intuition: "window width", consider $h(x) = h$, $K(u) = \mathbb{I}[u \leq 1]$.

Parameters

- Typically used $K(u)$[4]:

$$\begin{aligned} K_G(u) &= e^{-\frac{1}{2}u^2} - \text{Gaussian kernel} \\ K_P(u) &= (1-u^2)^2\mathbb{I}[|u|<1] - \text{quartic kernel} \end{aligned}$$

- Typically used $h(x)$:
    - $h(x) = const$
    - $h(x) = \rho(x, x_{i_K})$, where x_{i_K} - K-th nearest neighbour.
        - better for unequal distribution of objects

[4] Compare them in terms of required computation.

Solution

$$Q(\widehat{y}|x) = \sum_{i=1}^{N} w_i(x)(\widehat{y} - y_i)^2 \to \min_{\alpha \in \mathbb{R}}$$

$$w_i(x) = K\left(\frac{\rho(x, x_i)}{h(x)}\right)$$

- From stationarity condition $\frac{\partial Q}{\partial \widehat{y}} = 0$ obtain optimal $\widehat{y}(x)$:

$$\widehat{y}(x) = \frac{\sum_{i=1}^{N} y_i w_i(x)}{\sum_{i=1}^{N} w_i(x)} = \frac{\sum_{i=1}^{N} y_i K\left(\frac{\rho(x,x_i)}{h(x)}\right)}{\sum_{i=1}^{N} K\left(\frac{\rho(x,x_i)}{h(x)}\right)}$$

Comments

- Under general regularity conditions $\widehat{y}(x) \xrightarrow{P} E[y|x]$
- The specific form of the kernel function does not affect the accuracy much.
    - but may affect efficiency[5]
- Compared to K-NN: may use all objects, bandwidth controls smoothness.
    - under what selection of $K(u)$ and $h(x)$ it reduces to basic K-NN?

[5] how?

Comments

Insead of optimizing local constant $\widehat{y}$

$$Q(\widehat{y}|x) = \sum_{i=1}^{N} w_i(x)(\widehat{y} - y_i)^2 \to \min_{\alpha \in \mathbb{R}}$$

we could have optimized **local linear regression**

$$Q(\widehat{\beta}|x) = \sum_{i=1}^{N} w_i(x)(x^T\beta - y_i)^2 \to \min_{\alpha \in \mathbb{R}}$$

This better handles approximation on the edges of domain.

Table of Contents

1. Linear regression
2. Nonlinear dependencies
3. Regularization & restrictions.
4. Different loss-functions
5. Weighted account for observations
6. Nadaraya-Watson regression
7. Other types of regression

Linear monotonic regression

- We can impose restrictions on coefficients such as non-negativity:

$$\begin{cases} Q(\beta) = ||X\beta - Y||^2 \rightarrow \min_\beta \\ \beta_i \geq 0, \quad i = 1, 2, ...D \end{cases}$$

- Examples:
  - in credit scoring we know that salary should be positively correlated with credibility.
  - avaraging of forecasts of different prediction algorithms ($\beta_i = 0$ means, that i-th component does not improve accuracy of forecasting)

Support vector regression

Idea: don't care about small deviations, catch only the large ones + regularization.

$$\begin{cases} \frac{1}{2}\|w\|^2 \to \min_w \\ \langle w, x_n \rangle + w_0 - y_n \le \varepsilon & n = \overline{1, N} \\ y_n - \langle w, x_n \rangle - w_0 \le \varepsilon & n = \overline{1, N} \end{cases}$$

Since fitting any dataset with error$\in [-\varepsilon, \varepsilon]$ may be infeasible use penalization of excessive deviations:

$$\begin{cases} \frac{1}{2}\|w\|^2 + C\sum_{n=1}^{N}(\xi_n + \xi_n^*) \to \min_{w,\xi_n,\xi_n^*} \\ \langle w, x_n \rangle + w_0 - y_n \le \varepsilon + \xi_n, \quad \xi_n \ge 0 & n = \overline{1, N} \\ y_n - \langle w, x_n \rangle - w_0 \le \varepsilon + \xi_n^*, \quad \xi_n^* \ge 0 & n = \overline{1, N} \end{cases}$$

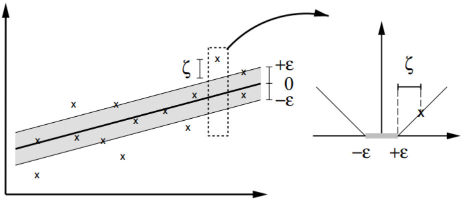
C controls how much errors should matter more than model simplicity.

Support vector regression

Equivalent unconstrained formulation:

$$\frac{1}{2}\|w\|^2 + C\sum_{n=1}^{N}\mathcal{L}(\langle w, x_n\rangle + w_0 - y_n) \to \min_w$$

with ε insensitive loss $\mathcal{L}(u) = \begin{cases} 0, & \text{if } |u| \le \varepsilon \\ |u| - \varepsilon & \text{otherwise} \end{cases}$



Solution will depend only on objects with $|\text{error}| \geq \varepsilon$, called *support vectors*.

Summary

- Linear regression gives interpretable analytic solution.
- Non-linear dependencies can be modelled by adding non-linear features.
- When features are linearly dependent, it fails.
- Regularized versions are always preferrable:
  - work in case of linearly dependent features
  - are more robust in close to linear dependence case
  - λ gives a convenient way to control model complexity
- Robust regression is robust to outliers.
  - we may also use robust loss-functions instead of MSE.