EECS 553: Machine Learning (ECE) **University of Michigan**

Linear Regression

Winter 2023 **Clayton Scott**

1 Regression

Regression is the other main supervised learning problem besides classification. Adopting a probabilistic perspective as we did for classification, there are jointly distributed variables $(\boldsymbol{X}, Y)$ where

$$\boldsymbol{X} \in \mathbb{R}^d, \qquad Y \in \mathbb{R}$$

and the goal is to predict Y from $\boldsymbol{X}$ using a function $f : \mathbb{R}^d \to \mathbb{R}$. In practice we don't have access to the joint distribution and must estimate the optimal f using training data $(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n)$.

A *regression model* is a collection of candidates for f. In these notes we will focus on the linear model where $f(\boldsymbol{x}) = \boldsymbol{w}^T\boldsymbol{x} + b$ for some $\boldsymbol{w} \in \mathbb{R}^d$, $b \in \mathbb{R}$. See Fig. 1 for an example.

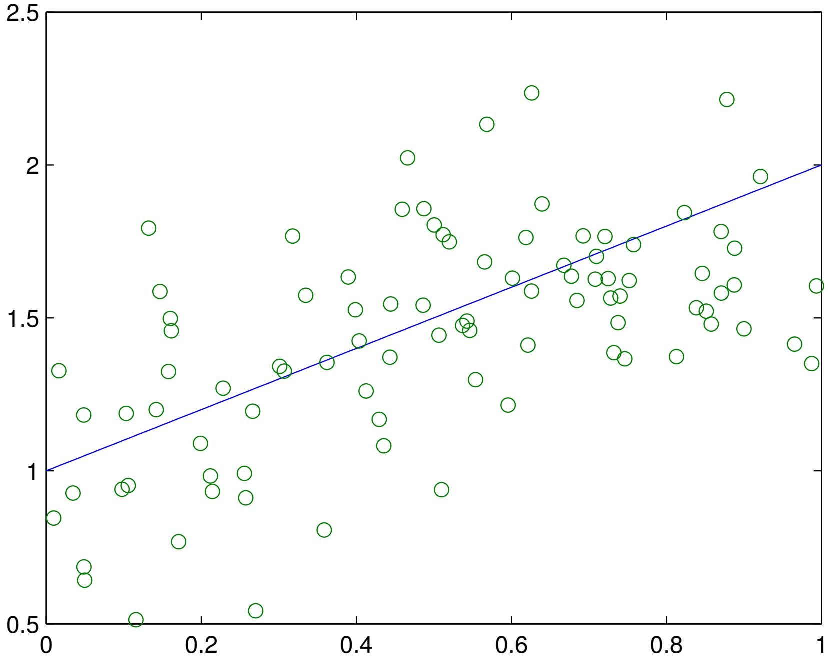


Figure 1: An example of linear regression

2 Mean Squared Error

Before focusing on linear regression, we first study the best regression function f given knowledge of the joint distribution of $\boldsymbol{X}$ and Y, parallel to our study of the Bayes classifier in classification. As a performance measure, we define *mean squared error* (MSE) of a regression function $f : \mathbb{R}^d \to \mathbb{R}$ to be

$$R(f) := E_{\boldsymbol{X}Y}[(f(\boldsymbol{X}) - Y)^2].$$

Just like classification, there is a regression function f^* that achieves the minimum value R^* of the mean squared error.

Theorem 1. *The function*

$$f^*(\boldsymbol{x}) := E_{Y|\boldsymbol{X}}[Y|\boldsymbol{X} = \boldsymbol{x}]$$

minimizes the mean squared error.

The function f^* is called the *conditional mean* predictor.As in classification, it depends only on the conditional distribution of Y given $\boldsymbol{X}$, and not on the marginal distribution of $\boldsymbol{X}$.

Proof. Let f be any regression function. Then

$$\begin{aligned}
R(f) &= E_{\boldsymbol{X}Y}[(f(\boldsymbol{X}) - Y)^2] \\
&= E_{\boldsymbol{X}} E_{Y|\boldsymbol{X}}[(f(\boldsymbol{X}) - Y)^2 | \boldsymbol{X}] \\
&= E_{\boldsymbol{X}} E_{Y|\boldsymbol{X}}[(f(\boldsymbol{X}) - E[Y|\boldsymbol{X}] + E[Y|\boldsymbol{X}] - Y)^2 | \boldsymbol{X}] \\
&= E_{\boldsymbol{X}} E_{Y|\boldsymbol{X}}[(f(\boldsymbol{X}) - E[Y|\boldsymbol{X}])^2] + (E[Y|\boldsymbol{X}] - Y)^2 \\
&\qquad - 2(f(\boldsymbol{X}) - E[Y|\boldsymbol{X}])(E[Y|\boldsymbol{X}] - Y)|\boldsymbol{X}]
\end{aligned}$$

The second term is independent of f, and the third term is zero. The first term can be made to equal 0 by taking f to be the conditional mean, so this minimizes the MSE. □

In practice, the joint distribution is not available to us, and so we instead attempt to minimize the empirical MSE,

$$\widehat{R}(f) := \frac{1}{n} \sum_{i=1}^{n} (y_i - f(\boldsymbol{x}_i))^2.$$

This objective can be motivated from two perspectives. The first is simply that it uses the law of large numbers to approximate an expectation with its sample average. The second perspective is to adopt a plug-in approach as we did for classification. In this case, we see from the theorem that the optimal predictor depends on the conditional distribution of Y given $\boldsymbol{X} = \boldsymbol{x}$. Therefore we adopt a partial statistical model for this conditional distribution. If we assume that $Y|\boldsymbol{X} = \boldsymbol{x}$ is Gaussian with mean $\boldsymbol{w}^T\boldsymbol{x} + b$ and variance σ^2, then maximum likelihood estimation of $\boldsymbol{w}$ and b leads to the same sum-of-squared-errors objective. The details of this derivation are left as an exercise.

Also note that f^* is typically not linear. For the rest of these notes, we will assume f is linear.[1] Later in the course we will study nonlinear methods for regression.

3 Least Squares Linear Regression

A linear regression estimate has the form $f(\boldsymbol{x}) = \boldsymbol{w}^T\boldsymbol{x} + b$ for some $\boldsymbol{w} \in \mathbb{R}^d$ and $b \in \mathbb{R}$. In the case of linear regression, we write

$$R(\boldsymbol{w}, b) = \mathbb{E}_{\boldsymbol{X},Y}\left[(Y - \boldsymbol{w}^T\boldsymbol{X} - b)^2\right]$$

for the MSE. Although the joint distribution of $(\boldsymbol{X}, Y)$ is unknown, we can estimate it via

$$\widehat{R}(\boldsymbol{w}, b) = \frac{1}{n} \sum_{i=1}^{n} \left(y_i - \boldsymbol{w}^T\boldsymbol{x}_i - b\right)^2.$$

[1] As with classification, a better term would be “affine” instead of “linear”.

Adding a regularization term for greater generality, a regression estimate is obtained by solving

$$\min_{\boldsymbol{w},b} \frac{1}{n} \sum_{i=1}^{n} \left(y_i - \boldsymbol{w}^T \boldsymbol{x}_i - b\right)^2 + \lambda \left\|\boldsymbol{w}\right\|^2 .$$

When $\lambda = 0$ the method is called *least squares regression* or *ordinary least squares.* For $\lambda > 0$ it is called *ridge regression* and the term $\lambda\|\boldsymbol{w}\|^2$ is called the *ridge penalty.* We will now derive the solution in two ways.

3.1 Solution 1: Eliminate b

We can eliminate b by requiring the partial derivative with respect to b to be 0. This yields the equation

$$-\frac{2}{n} \sum_{i=1}^{n} \left(y_i - \boldsymbol{w}^T \boldsymbol{x}_i - b\right) = 0$$

whose solution is

$$\begin{aligned} b &= \frac{1}{n} \sum_{i=1}^{n} (y_i - \boldsymbol{w}^T \boldsymbol{x}_i) \\ &= \bar{y} - \boldsymbol{w}^T \bar{\boldsymbol{x}} \end{aligned}$$

where $\bar{y} = \frac{1}{n}\sum_i y_i$, $\bar{\boldsymbol{x}} = \frac{1}{n}\sum_i \boldsymbol{x}_i$. Plugging this in, the objective function becomes

$$\frac{1}{n} \sum_{i=1}^{n} \left[y_i - \bar{y} - \boldsymbol{w}^T(\boldsymbol{x}_i - \bar{\boldsymbol{x}})\right]^2 + \lambda \left\|\boldsymbol{w}\right\|^2 .$$

Let's denote $\tilde{y}_i = y_i - \bar{y}$, $\tilde{\boldsymbol{x}}_i = \boldsymbol{x}_i - \bar{\boldsymbol{x}}$.

Next, observe

$$\sum_{i=1}^{n} \left(\tilde{y}_i - \boldsymbol{w}^T \tilde{\boldsymbol{x}}_i\right)^2 = \left\|\tilde{\boldsymbol{y}} - \tilde{\boldsymbol{X}}\boldsymbol{w}\right\|^2$$

where

$$\tilde{\boldsymbol{y}} = \begin{bmatrix} \tilde{y}_1 \\ \vdots \\ \tilde{y}_n \end{bmatrix} \qquad \tilde{\boldsymbol{X}} = \begin{bmatrix} \tilde{x}_{11} & \cdots & \tilde{x}_{1d} \\ \vdots & \ddots & \vdots \\ \tilde{x}_{n1} & \cdots & \tilde{x}_{nd} \end{bmatrix} .$$

Therefore the objective function is

$$\begin{aligned} \frac{1}{n} \left\|\tilde{\boldsymbol{y}} - \tilde{\boldsymbol{X}}\boldsymbol{w}\right\|^2 + \lambda \left\|\boldsymbol{w}\right\|^2 &\propto \left(\tilde{\boldsymbol{y}} - \tilde{\boldsymbol{X}}\boldsymbol{w}\right)^T \left(\tilde{\boldsymbol{y}} - \tilde{\boldsymbol{X}}\boldsymbol{w}\right) + n\lambda \boldsymbol{w}^T \boldsymbol{w} \\ &= \boldsymbol{w}^T \left(\tilde{\boldsymbol{X}}^T \tilde{\boldsymbol{X}} + n\lambda \boldsymbol{I}\right) \boldsymbol{w} - 2\tilde{\boldsymbol{y}}^T \tilde{\boldsymbol{X}} \boldsymbol{w} + \tilde{\boldsymbol{y}}^T \tilde{\boldsymbol{y}} \\ &= \frac{1}{2} \boldsymbol{w}^T \boldsymbol{A} \boldsymbol{w} + \boldsymbol{r}^T \boldsymbol{w} + c \end{aligned}$$

where $\boldsymbol{A} = 2(\tilde{\boldsymbol{X}}^T \tilde{\boldsymbol{X}} + n\lambda \boldsymbol{I})$, $\boldsymbol{r} = -2\tilde{\boldsymbol{X}}^T \tilde{\boldsymbol{y}}$, and $c = \tilde{\boldsymbol{y}}^T \tilde{\boldsymbol{y}}$. Notice that $\boldsymbol{A} \succeq \boldsymbol{0}$ (positive semi-definite), and $\boldsymbol{A} \succ 0$ if $\lambda > 0$.

Denote the objective

$$J(\boldsymbol{w}) := \frac{1}{2} \boldsymbol{w}^T \boldsymbol{A} \boldsymbol{w} + \boldsymbol{r}^T \boldsymbol{w} + c.$$

We know from the theory of unconstrained optimization that $\boldsymbol{w}^*$ is a global minimizer of J iff

$$\nabla J(\boldsymbol{w}^*) = \boldsymbol{A}\boldsymbol{w}^* + \boldsymbol{r} = \boldsymbol{0}.$$

If $\boldsymbol{A}$ is invertible, which occurs if $\boldsymbol{A} \succ \boldsymbol{0}$, then

$$\boldsymbol{w}^* = -\boldsymbol{A}^{-1}\boldsymbol{r} = \left(\tilde{\boldsymbol{X}}^T\tilde{\boldsymbol{X}} + n\lambda\boldsymbol{I}\right)^{-1}\tilde{\boldsymbol{X}}^T\tilde{\boldsymbol{y}}$$

is the *unique* global minimizer.

3.2 Solution 2: Don't Eliminate b

It is also possible to drive the solution without first eliminating b. Denote $\boldsymbol{\theta} = [b, \boldsymbol{w}^T]^T$. Similar to the previous case we have

$$\sum_{i=1}^{n}\left(y_i - \boldsymbol{w}^T\boldsymbol{x}_i - b\right)^2 = \|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\theta}\|^2$$

where now

$$\boldsymbol{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} \qquad \boldsymbol{X} = \begin{bmatrix} 1 & x_{11} & \cdots & x_{1d} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n1} & \cdots & x_{nd} \end{bmatrix}.$$

The overall objective can be expressed

$$J(\boldsymbol{\theta}) := \boldsymbol{\theta}^T(\boldsymbol{X}^T\boldsymbol{X} + n\lambda\boldsymbol{I}^0_{d+1})\boldsymbol{\theta} - 2\boldsymbol{y}^T\boldsymbol{X}\boldsymbol{\theta} + \boldsymbol{y}^T\boldsymbol{y}$$

where $\boldsymbol{I}^0_{d+1}$ is the $(d+1) \times (d+1)$ matrix obtaining by taking the identity matrix and setting the first diagonal entry to 0.

Arguing as above, we find

$$\boldsymbol{\theta}^* = (\boldsymbol{X}^T\boldsymbol{X} + n\lambda\boldsymbol{I}^0_{d+1})^{-1}\boldsymbol{X}^T\boldsymbol{y}$$

provided $\boldsymbol{X}^T\boldsymbol{X} + n\lambda\boldsymbol{I}^0_{d+1}$ is invertible. As an exercise, you are asked to show that this is the case whenever $\lambda > 0$.

4 Large Scale Ridge Regression

This section focuses on computational issues and least-squares regression in high dimensions. We focus on the first derivation although similar comments apply to the second.

Even though least-squares regression has a closed form solution, the exact formula can be computationally prohibitive because of the need to invert a $d \times d$ matrix. An alternative is to minimize

$$J(\boldsymbol{w}) = \boldsymbol{w}^T\left(\tilde{\boldsymbol{X}}^T\tilde{\boldsymbol{X}} + n\lambda\boldsymbol{I}\right)\boldsymbol{w} - 2\tilde{\boldsymbol{y}}^T\tilde{\boldsymbol{X}}\boldsymbol{w} + \tilde{\boldsymbol{y}}^T\tilde{\boldsymbol{y}}$$

using an iterative algorithm such as gradient descent.

The gradient is

$$\begin{aligned}\nabla J(\boldsymbol{w}) =& 2(\tilde{\boldsymbol{X}}^T\tilde{\boldsymbol{X}} + n\lambda\boldsymbol{I})\boldsymbol{w} - 2\tilde{\boldsymbol{X}}^T\tilde{\boldsymbol{y}} \\ =& 2\tilde{\boldsymbol{X}}^T(\tilde{\boldsymbol{X}}\boldsymbol{w} - \tilde{\boldsymbol{y}}) + 2n\lambda\boldsymbol{I}\boldsymbol{w} \\ =& 2\sum_{i=1}^{n}\left[\tilde{\boldsymbol{x}}_i(\boldsymbol{w}^T\boldsymbol{x}_i - \tilde{y}_i) + \lambda\boldsymbol{w}\right]. \end{aligned} \tag{1}$$

The computational complexity of calculating the gradient is $O(nd)$ per iteration, which makes gradient descent much more scalable than applying the exact formula.

Because the gradient can be expressed as a summation over the training data as in (1), another option is stochastic gradient descent. SGD can converge much faster than gradient descent, and is particularly useful when the full gradient is expensive to compute/store. An extension of gradient descent called *conjugate gradient descent* often offers very good performance.

5 Robust Regression

Least squares has some disadvantages. Consider the data shown in Fig. 2: Least squares is not robust to outliers because errors get squared, which gives outliers too much influence on the final solution.

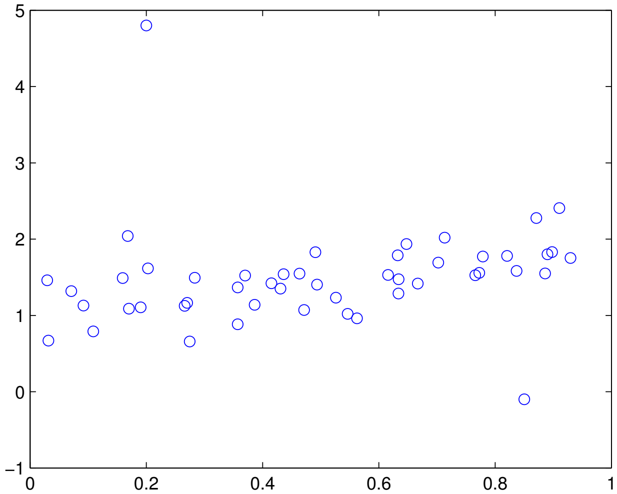


Figure 2: Least squares is not robust to outliers.

An alternative is to solve

$$\min_{\boldsymbol{w},b} \frac{1}{n}\sum_{i=1}^{n} \rho(y_i - \boldsymbol{w}^T\boldsymbol{x}_i - b)$$

where ρ is a *robust loss function.* See Fig. 3 for some examples. Except for the squared error loss in (a), these examples all grow at most linearly with the prediction error. Note that some ρ are convex, which is good from an optimization standpoint, whereas others are not, which is better from a robustness standpoint. The tradeoff between convexity and robustness is a common theme in machine learning.

Unlike least squares, there is no closed form solution. To minimize

$$J(\boldsymbol{w}, b) = \frac{1}{n}\sum_{i=1}^{n} \rho(y_i - \boldsymbol{w}^T\boldsymbol{x}_i - b)$$

numerically, one could apply gradient descent or SGD, although a majorize/minimize (MM) algorithm is often used.

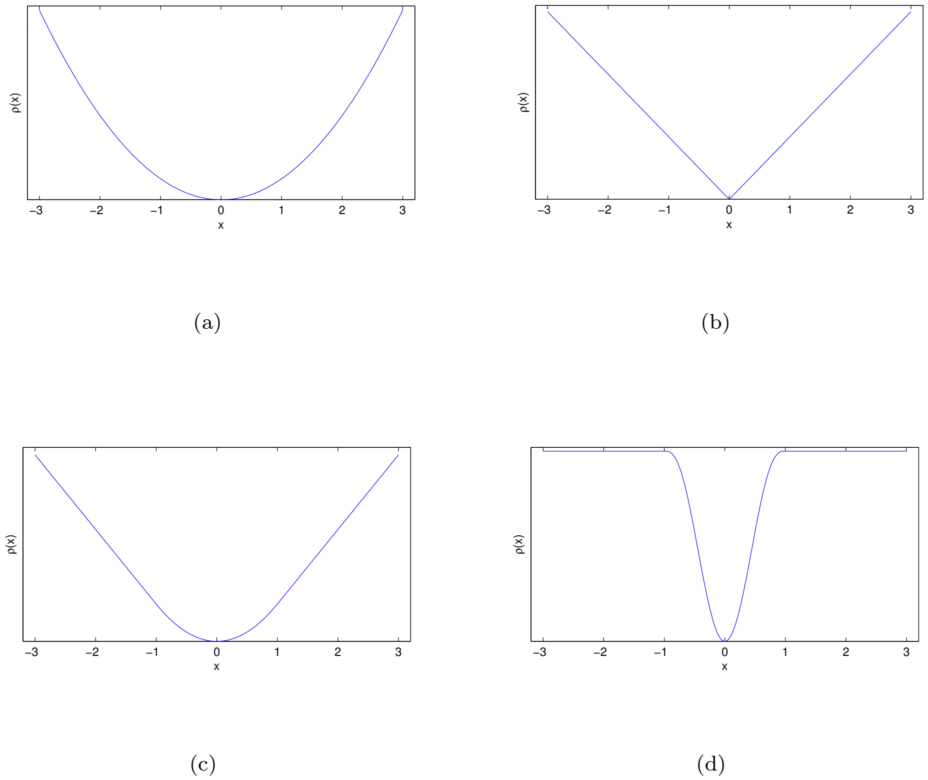
In the MM approach, the idea is to choose a majorizing function J_t at every iteration, and to minimize that function instead of J. For J_t to be a majorizing function, it needs to upper bound $J(\boldsymbol{\theta})$ and satisfy $J_t(\boldsymbol{\theta}_t) = J(\boldsymbol{\theta}_t)$. It should also be easily minimized with respect to $\boldsymbol{\theta}$. We will select J_t to have the form

$$J_t(\boldsymbol{w}, b) = \sum_{i=1}^{n} \rho_t(y_i - \boldsymbol{w}^T\boldsymbol{x}_i - b)$$

where ρ_t is a majorizing function for ρ.

Let us introduce the notation

$$\psi(r) := \rho'(r)$$
$$\varphi(r) := \frac{\psi(r)}{r} \qquad r \neq 0$$



(a)

(b)

(c)

(d)

Figure 3: (a) Squared errors; (b) Absolute errors; (c) Winsorizing at 1.5 (a.k.a. Huber loss); (d) Biweight.

and

$$r_{t,i} = y_i - \boldsymbol{w}_t^T \boldsymbol{x}_i - b_t$$

The following result provides a majorizing function for a broad class of ρ.

Lemma 1. *Assume that $\rho(r)$ is symmetric, differentiable, and nondecreasing for $r > 0$, that $\psi(r)/r$ is nonincreasing for $r > 0$, that $\varphi(0) := \lim_{r \to 0} \varphi(r)$ exists, and that $\varphi(r)$ is continuous. Define*

$$J_t(\boldsymbol{w}, b) = \sum_{i=1}^{n} \rho_t(y_i - \boldsymbol{w}^T \boldsymbol{x}_i - b)$$

where

$$\rho_t(r) = \rho(r_{t,i}) - \frac{1}{2} r_{t,i} \psi(r_{t,i}) + \frac{1}{2} \frac{\psi(r_{t,i})}{r_{t,i}} r^2.$$

Then J_t majorizes J.

Proof. The proof is left as an exercise. The conditions of the lemma are satisfied by Huber's ρ and the biweight function, as well as several others. □

With this majorizing function, the iterative update has the form

$$(\boldsymbol{w}_{t+1}, b_{t+1}) = \arg\min_{\boldsymbol{w}, b} \sum_{i=1}^{n} c_{t,i} (y_i - \boldsymbol{w}^T \boldsymbol{x}_i - b)^2$$

where

$$c_{t,i} = \frac{\psi(r_{t,i})}{r_{t,i}} = \varphi(r_{t,i}).$$

The algorithm is known as *iteratively reweighted least squares*. Weighted least squares can be solved by a slight modification of ordinary least squares. This adaptation is left as an exercise.

Exercises

1. (★) Give an example of a training dataset where $d = 2$ and $n = 3$, but ordinary least squares does not have a unique solution.

2. (★) Consider the second derivation of least squares regression. Fill in the missing steps to complete the derivation.

3. Show that the following matrices are positive semi-definite, and positive definite when $\lambda > 0$:

   (a) (★) $\tilde{\boldsymbol{X}}^T \tilde{\boldsymbol{X}} + n\lambda \boldsymbol{I}_d$

   (b) (★★) $\boldsymbol{X}^T \boldsymbol{X} + n\lambda \boldsymbol{I}_{d+1}^0$

4. This problem motivates the need for regularization in high dimensions.

   (a) (★★) For any $p \times q$ matrix $\boldsymbol{B}$, with $q \leq p$, argue that $\boldsymbol{B}^T \boldsymbol{B}$ is invertible iff $\boldsymbol{B}$ has full rank, i.e., the columns of $\boldsymbol{B}$ are linearly independent.

   (b) (★) Explain why regularization ($\lambda > 0$) is necessary when $d > n$.

5. (★) Verify Equation (1). *Hint:* Recall that a matrix times a column vector can be viewed as a linear combination of the columns of the matrix, where the weights are the entries of the column vector.

6. (★) Determine a decomposition for the gradient with respect to $\boldsymbol{\theta}$, similar to (1), for the least squares objective according to the derivation in 3.2.

7. Consider linear regression and let $c_1, \ldots, c_n > 0$ be known weights. Determine the solution of

$$\min_{\boldsymbol{w}, b} \sum_{i=1}^{n} c_i (y_i - \boldsymbol{w}^T \boldsymbol{x}_i - b)^2.$$

Express your solution in terms of the matrix $\boldsymbol{C} = \mathrm{diag}(c_1, \ldots, c_n)$ an appropriate data matrix $\boldsymbol{X}$, and other notation as needed.

Hint: As a sanity check, your solution should reduce to the ordinary least squares solution when C is the identity matrix.

8. (★) Use the previous problem to concisely state the IRLS algorithm for robust linear regression.

9. This problem develops a maximum likelihood approach to linear regression. Let $(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n)$ be training data for a regression problem, where $\boldsymbol{x}_i \in \mathbb{R}^d$ and $y_i \in \mathbb{R}$. Let $f(\boldsymbol{x})$ denote the regression model, and assume

$$y_i = f(\boldsymbol{x}_i) + \epsilon_i$$

for each i, where

$$\epsilon_i \sim \mathcal{N}(0, \sigma^2)$$

and the ϵ_i are independent. Also, assume $f(\boldsymbol{x}) = \boldsymbol{w}^T \boldsymbol{x} + b$. From this statistical model, the parameters $\boldsymbol{w}$ and b may be estimated by maximum likelihood estimation (MLE). Notice that the input variables $\boldsymbol{x}_i$ are considered fixed and nonrandom, so that the likelihood is actually a conditional likelihood, similar to logistic regression.

   (a) (★★) Show that the maximum likelihood estimate of $(\boldsymbol{w}, b)$ coincides with the ordinary least squares linear regression estimate developed in class.

   (b) (★★) Now suppose that the noise variance is different for each data point,

$$\epsilon_i \sim \mathcal{N}(0, \sigma_i^2).$$

   a type of noise called *heteroscedastic* noise. (When all the variances are the same it's called *homoscedastic.*) Assume the variances σ_i^2 are known (although in practice they might also need to be estimated). Explain how to compute the MLE in this situation.

10. (★★) In this problem you will develop a method for nonlinear regression called *locally linear regression.* Let $(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n)$ be training data for a regression problem, with $\boldsymbol{x}_i \in \mathbb{R}^d$ and $y_i \in \mathbb{R}$. Let $k(\boldsymbol{x}, \boldsymbol{x}')$ be a *local weighting kernel*, meaning a function that outputs nonnegative values and is nonincreasing as a function of $\|\boldsymbol{x} - \boldsymbol{x}'\|$. The Gaussian kernel $k(\boldsymbol{x}, \boldsymbol{x}') = \exp(-\|\boldsymbol{x} - \boldsymbol{x}'\|^2 / 2\sigma^2)$ is an example.

The idea behind locally linear regression is to fit a linear regression model using a weighted least-squares criterion at each test point, where the weights are determine by the local weighting kernel. In particular, the predicted value is y at a test point $\boldsymbol{x}$ is

$$y = \boldsymbol{w}_{\boldsymbol{x}}^T \boldsymbol{x} + b_{\boldsymbol{x}}$$

where $\boldsymbol{w}_{\boldsymbol{x}}$ and $b_{\boldsymbol{x}}$ are the solution of

$$\min_{\boldsymbol{w}, b} \sum_{i=1}^{n} k(\boldsymbol{x}, \boldsymbol{x}_i)(y_i - \boldsymbol{w}^T \boldsymbol{x}_i - b)^2.$$

Determine a closed form expression for the nonlinear regression estimate $\widehat{f}(\boldsymbol{x}) := \boldsymbol{w}_{\boldsymbol{x}}^T \boldsymbol{x} + b_{\boldsymbol{x}}$ as a function of $\boldsymbol{x}$.

11. (☆☆☆) Prove Lemma 1 by proceeding as follows. Consider the function

$$m(r) := \rho_t(r) - \rho(r).$$

You need to show $m(r_{t,i}) = 0$, and that $m(r) \geq 0$ for all $r \in \mathbb{R}$.

(a) Verify that $m(r_{t,i}) = m(-r_{t,i}) = 0$.

(b) Verify that $m'(r_{t,i}) = m'(-r_{t,i}) = 0$.

(c) In the case where $r_{t,i} > 0$, argue that

$$m'(r) \begin{cases} \leq 0, & 0 < r \leq r_{t,i}, \\ \geq 0, & r_{t,i} \leq r. \end{cases}$$

(d) By considering different cases for $r_{t,i}$, and with an appeal to symmetry, conclude that $m(r) \geq 0$ for all r.

Note: I'm not 100% certain the conditions in the lemma are exactly the right ones, so if you find that they can be weakened, or need to be strengthened, that's fine.

12. (☆☆) Let $\boldsymbol{\theta}$ encompass the parameters $\boldsymbol{w}$ and b for robust linear regression, and let $\tilde{\boldsymbol{x}}_i$ denote $\boldsymbol{x}_i$ with a constant one appended. Let J be the robust linear regression objective function. Assuming ρ is differentiable, the gradient of J is

$$\nabla J(\boldsymbol{\theta}) = -\sum_{i=1}^{n} \psi(r_i(\boldsymbol{\theta}))\tilde{\boldsymbol{x}}_i.$$

where $r_i(\boldsymbol{\theta}) = y_i - \tilde{\boldsymbol{x}}_i^T \boldsymbol{\theta}$.

(a) Argue that if $\boldsymbol{\theta}^*$ is a local minimum of J, then

$$\boldsymbol{\theta}^* = f(\boldsymbol{\theta}^*)$$

for a certain function f. *Hint*: Insert the quantity $\frac{r_i(\boldsymbol{\theta})}{r_i(\boldsymbol{\theta})}$ into the expression for the gradient.

(b) Use the previous result to provide an alternate interpretation of IRLS, which we originally derived from a majorize-minimize perspective. Namely, show that the IRLS algorithm is attempting to find a $\boldsymbol{\theta}^*$ satisfying the necessary condition by finding a fixed point of f using fixed point iteration (look up fixed point iteration if you're not familiar with it).