EECS545 Machine Learning
Homework #1 Solutions

Notation: As in the lecture, we will interchangeably use $\mathbf{x}_i$ and $\mathbf{x}^{(i)}$ to denote the i-th training example (and similar notation for other variables). Often, it's less confusing to use $\mathbf{x}^{(i)}$ than $\mathbf{x}_i$ (especially in handwriting) when your expression involves another subscripts to denote specific coordinates in the vector. The Bishop's book uses the former notation, but some other machine learning textbooks use the latter notation. You are free to use whichever notation as long as you are clear about it.

1 [42 points] Linear regression on a polynomial

(a) **[15 points]**

i. **Answer[12 points]:** These solutions are approx. and exact solutions depend on the learning rates, the convergence criterion used (e.g., norm of the gradient is small, reduction in objective value is small, etc.). The following image is not required, just instructive.

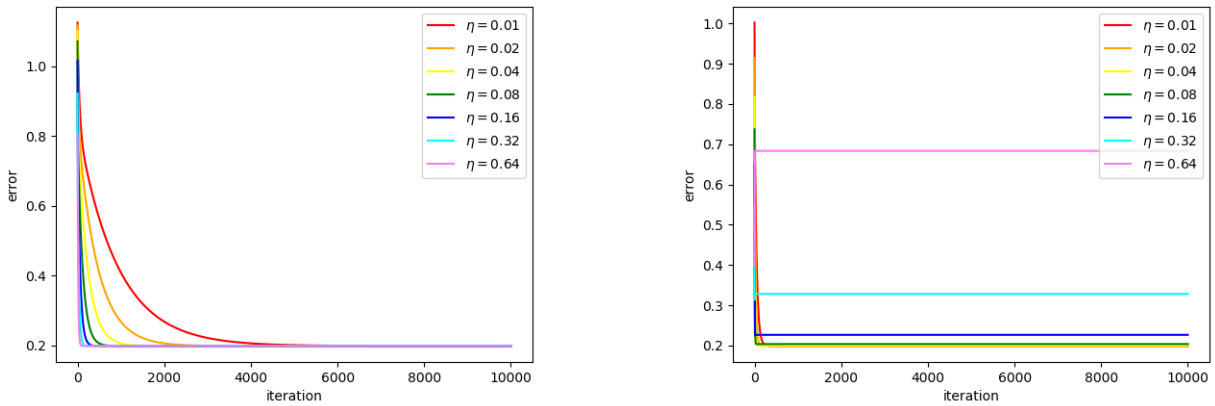


Figure 1: Learning curve for the batch gradient method, where x axis is training iteration and y axis is mean squared error on the training dataset. The different colors represents different learning rates.

See attached `q1_sol.py`.

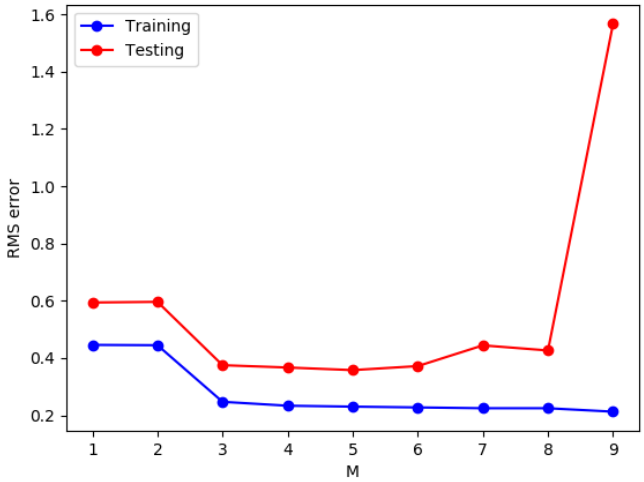
- Batch gradient descent: $y = 1.9469 - 2.8242x$ (with learning rate 0.08 after 10000 iterations, BGD achieves the smallest MSE 0.1988)
- Stochastic gradient descent: $y = 1.9429 - 2.8298x$ (with learning rate 0.01 after 10000 iterations, SGD achieves the smallest MSE 0.1988)

ii. **Answer[3 points]:** With learning rate 0.64, BGD converges within 185 iterations (i.e. keeping MSE lower than 0.02 for 100 iterations). With learning rate 0.04, SGD converges within 177 iterations (i.e. keeping MSE lower than 0.02 for 100 iterations). In terms of number of "outerloop" iterations,

stochastic gradient descent converges slightly more quickly than the batch gradient descent in this question. [Note: with different implementation details, different learning rates or different convergence criteria, the number of iterations for convergence may be different. As long as the answer is reasonable and consistent with the output from your code, we will not deduct points.]

(b) **[15 points]**

i. **Answer[10 points]:**



ii. **Answer[5 points]:** For the figure above, $M = 5$ minimizes the test error (while also performing reasonably well on training error).

(c) **[12 points]**

i. **Answer[10 points]:** See the plots below.

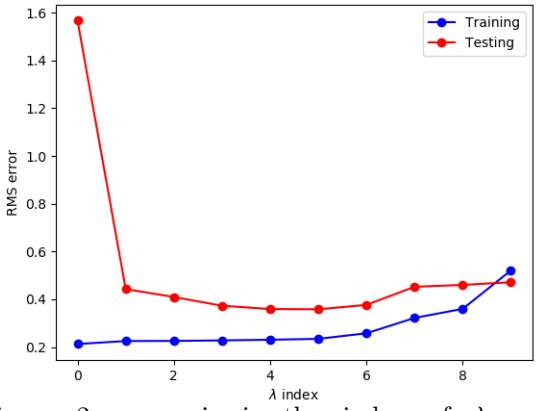


Figure 2: x axis is the index of $\lambda = \{0, 10^{-8}, 10^{-7}, 10^{-6}, 10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 10^{0}\}$, y axis is error on the training and test dataset.

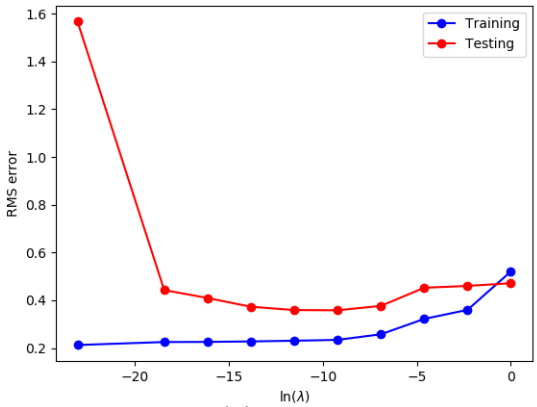


Figure 3: x axis is $\ln(\lambda)$ value and y axis is error on the training and test dataset.

ii. **Answer[2 points]:** $\lambda = 10^{-5}..10^{-4}$ seems to be the "sweet spot" for this particular problem.

2 [36 points] Locally weighted linear regression

(a) **[2 points]** Let $\mathbf{z} = X\mathbf{w} - \mathbf{y}$, i.e. $z_i = \mathbf{w^T x}_i - y_i$. Then we have:

$$\begin{aligned} E_D(\mathbf{w}) &= \frac{1}{2}\sum_{i=1}^{N} r_i(\mathbf{w^T x}_i - y_i)^2 \\ &= \sum_{i=1}^{N} \frac{1}{2} r_i z_i^2 \\ &= \mathbf{z}^T R \mathbf{z} \\ &= (X\mathbf{w} - \mathbf{y})^T R (X\mathbf{w} - \mathbf{y}) \end{aligned}$$

where $R_{ii} = \frac{1}{2} r_i$, $R_{ij} = 0$ for $i \neq j$.

(b) **[8 points]**

$$\nabla_{\mathbf{w}} E_D(\mathbf{w}) = \nabla_{\mathbf{w}} (\mathbf{w}^T X^T R X \mathbf{w} + \mathbf{y}^T R \mathbf{y} - 2\mathbf{y}^T R X \mathbf{w}) = 2X^T R X \mathbf{w} - 2X^T R \mathbf{y}$$

So, $\nabla_{\mathbf{w}} E_D(\mathbf{w}) = 0$ when

$$X^T R X \mathbf{w} = X^T R \mathbf{y}$$

These are the normal equations, from which we can get a closed form formula for $\mathbf{w}$:

$$\mathbf{w} = (X^T R X)^{-1} X^T R \mathbf{y}$$

(c) **Answer[8 points]:**

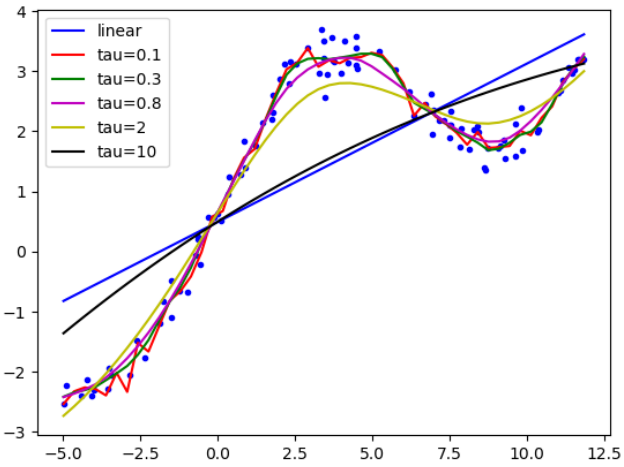
$$\begin{aligned} \arg\max_{\mathbf{w}} \prod_{i=1}^{N} p(y_i|\mathbf{x}_i; \mathbf{w}) &= \arg\max_{\mathbf{w}} \sum_{i=1}^{N} \log p(y_i|\mathbf{x}_i; \mathbf{w}) \\ &= \arg\max_{\mathbf{w}} \sum_{i=1}^{N} \log \frac{1}{\sqrt{2\pi}\sigma_i} - \frac{(y_i - \mathbf{w}^T \mathbf{x}_i)^2}{2(\sigma_i)^2} \\ &= \arg\min_{\mathbf{w}} \frac{1}{2} \sum_{i=1}^{N} \frac{(y_i - \mathbf{w}^T \mathbf{x}_i)^2}{(\sigma_i)^2} \\ &= \arg\min_{\mathbf{w}} \frac{1}{2} \sum_{i=1}^{N} \frac{1}{(\sigma_i)^2} (y_i - \mathbf{w}^T \mathbf{x}_i)^2 \\ &= \arg\min_{\mathbf{w}} \frac{1}{2} \sum_{i=1}^{N} r_i (y_i - \mathbf{w}^T \mathbf{x}_i)^2 \end{aligned}$$

where in the last step, we substituted $r_i = \frac{1}{(\sigma_i)^2}$ to get the linear regression form.

(d) **Answer[18 points]:**

See attached `q2_sol.py`.

For small bandwidth parameter τ, the fitting is dominated by the closest by training samples. The smaller the bandwidth, the less training samples that are actually taken into account when doing the regression, and the regression results thus become very susceptible to noise in those few training samples. For larger τ, we have enough training samples to reliably fit straight lines, unfortunately a straight line is not the right model for these data, so we also get a bad fit for large bandwidths.



3 [22 points] Derivation and Proof

(a) **[8 points]** Consider the 1D case of linear regression. The loss function is $L(w_0, w_1) = \sum_{i=1}^{N}(y^{(i)} - w_0 - w_1 x^{(i)})^2$.

The gradient for w_0 is $\frac{\partial L}{\partial w_0} = -2\sum_{i=1}^{N}(y^{(i)} - w_0 - w_1 x^{(i)})$. Let the gradient be 0. Then the solution for w_0 satisfies $\sum_{i=1}^{N}(y^{(i)} - w_0 - w_1 x^{(i)}) = \sum_{i=1}^{N} y^{(i)} - nw_0 - w_1 \sum_{i=1}^{N} x^{(i)} = 0$.

Therefore, the solution is $w_0 = \frac{\sum_{i=1}^{N} y^{(i)}}{N} - w_1 \frac{\sum_{i=1}^{N} x^{(i)}}{N} = \bar{Y} - w_1 \bar{X}$.

The gradient for w_1 is $\frac{\partial L}{\partial w_1} = -2\sum_{i=1}^{N} x^{(i)}(y^{(i)} - w_0 - w_1 x^{(i)})$. Let the gradient be 0. We have $\sum_{i=1}^{N}(x^{(i)}y^{(i)} - w_0 x^{(i)} - w_1 x^{(i)2}) = 0$.

Replace w_0 with its solution $\bar{Y} - w_1\bar{X}$.

We have $\sum_{i=1}^{N}(x^{(i)}y^{(i)} - (\bar{Y} - w_1\bar{X})x^{(i)} - w_1 x^{(i)2}) = 0$.

So $\sum_{i=1}^{N}(x^{(i)}y^{(i)} - \bar{Y}x^{(i)}) - w_1 \sum_{i=1}^{N}(x^{(i)2} - \bar{X}x^{(i)}) = 0$.

Therefore, the solution is $w_1 = \frac{\sum_{i=1}^{N}(x^{(i)}y^{(i)} - \bar{Y}x^{(i)})}{\sum_{i=1}^{N}(x^{(i)2} - \bar{X}x^{(i)})} = \frac{\sum_{i=1}^{N} x^{(i)}y^{(i)} - N\bar{Y}\bar{X}}{\sum_{i=1}^{N} x^{(i)2} - N\bar{X}^2}$.

(b) **[14 points]**

i. [6 points]
($\leftarrow$) Assume A is a positive definite matrix. Then for each row vector in $\mathbf{u}_i$ in $\mathbf{U}$, we have $\lambda_i = \lambda_i \mathbf{u}_i^T \mathbf{u}_i$ because $\mathbf{u}_i^T \mathbf{u}_i = 1$.

$$\lambda_i = \mathbf{u}_i^T(\lambda_i \mathbf{u}_i) = \mathbf{u}_i^T(A\mathbf{u}_i) > 0$$

.

($\rightarrow$) Assume $\lambda_i > 0$ for any i. Then for any $z \neq \mathbf{0}$, $z^T A z = z^T(U^T \Lambda U)z = z^T(\sum_{i=1}^{d} \lambda_i u_i u_i^T)z = \sum_{i=1}^{d} \lambda_i (z^T u_i)(u_i^T z) = \sum_{i=1}^{d} \lambda_i (z^T u_i)^2 \geq 0$.

ii. [8 points] Consider the real symmetric matrix $X^T X$. With any $z \in \mathcal{R}^d$, $z^T X^T X z = (Xz)^T(Xz) \geq 0$.
Therefore $X^T X$ is PSD, and we can have $X^T X = U\Lambda U^T = \sum_{i=1}^{d} \lambda_i u_i u_i^T$.
With ridge regression, we consider the matrix $X^T X + \beta \mathbf{I}$.
$X^T X + \beta \mathbf{I} = \sum_{i=1}^{d} \lambda_i u_i u_i^T + \beta \mathbf{I} = \sum_{i=1}^{d} \lambda_i u_i u_i^T + \beta \sum_{i=1}^{d} u_i u_i^T = \sum_{i=1}^{d} (\lambda_i + \beta) u_i u_i^T$.
Therefore, ridge regression has an effect of shifting all eigenvalues by a constant β. If $\beta > 0$, ridge regression makes the matrix $X^T X + \beta \mathbf{I}$ positive definite because all singular values are positive.