Homework 1

仲泽为 122090814

Due: Mar. 9 23:59

Q1 (1) $R(\alpha) = \begin{pmatrix} \cos(\alpha) & -\sin(\alpha) & 0 \\ \sin(\alpha) & \cos(\alpha) & 0 \\ 0 & 0 & 1 \end{pmatrix}$

$R^T = \begin{pmatrix} \cos(\alpha) & \sin(\alpha) & 0 \\ -\sin(\alpha) & \cos(\alpha) & 0 \\ 0 & 0 & 1 \end{pmatrix}$

$$R^TR = \begin{pmatrix} \cos(\alpha) & \sin(\alpha) & 0 \\ -\sin(\alpha) & \cos(\alpha) & 0 \\ 0 & 0 & 1 \end{pmatrix}\begin{pmatrix} \cos(\alpha) & -\sin(\alpha) & 0 \\ \sin(\alpha) & \cos(\alpha) & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

$$= \begin{pmatrix} \cos^2(\alpha)+\sin^2(\alpha) & -\cos(\alpha)\sin(\alpha)+\sin(\alpha)\cos(\alpha) & 0 \\ -\sin(\alpha)\cos(\alpha)+\sin(\alpha)\cos(\alpha) & \sin^2(\alpha)+\cos^2(\alpha) & 0 \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} = I_3$$

R is orthogonal

---

(2) For any orthogonal $Q \in \mathbb{R}^{n\times n}$, $Q^TQ = I_n$

Assume $\underline{x} \in \mathbb{R}^n$ is an eigenvector of Q with eigenvalue λ ($\lambda \in \mathbb{R}$)

i.e. $Q\underline{x} = \lambda\underline{x}$

$(Q\underline{x})^T Q\underline{x} = (\lambda\underline{x})^T\lambda\underline{x} \Rightarrow \lambda^2 \underline{x}^T\underline{x} = \underline{x}^TQ^TQ\underline{x} = \underline{x}^TI\underline{x} = \underline{x}^T\underline{x} \Rightarrow \lambda^2 = 1$

Thus, λ is either 1 or -1

Thus, eigenvalue of Q are either 1 or -1

Q2 (1) $\forall x, y \in \mathbb{R}, \quad \forall \theta \in [0,1]$

$f(\theta x + (1-\theta)y) = |\theta x + (1-\theta)y| \geq 0$

$\theta f(x) + (1-\theta)y = \theta|x| + (1-\theta)|y| \geq 0$

$[f(\theta x + (1-\theta)y)]^2 - [\theta f(x) + (1-\theta)y]^2$

$= |\theta x + (1-\theta)y|^2 - [\theta|x| + (1-\theta)|y|]^2$

$= \theta^2 x^2 + (1-\theta)^2 y^2 + 2\theta(1-\theta)xy - \theta^2|x|^2 - (1-\theta)^2|y|^2 - 2\theta(1-\theta)|x||y|$

$= 2\theta(1-\theta)(xy - |xy|)$

If $xy < 0 \Rightarrow [f(\theta x + (1-\theta)y)]^2 - [\theta f(x) + (1-\theta)y]^2 = 4\theta(1-\theta)xy < 0$

$\Rightarrow [f(\theta x + (1-\theta)y)]^2 < [\theta f(x) + (1-\theta)y]^2$

$\Rightarrow f(\theta x + (1-\theta)y) < \theta f(x) + (1-\theta)f(y)$

If $xy \geq 0 \Rightarrow [f(\theta x + (1-\theta)y]^2 - [\theta f(x) + (1-\theta)f(y)]^2 = 0$

$\Rightarrow [f(\theta x + (1-\theta)y)]^2 = [\theta f(x) + (1-\theta)y]^2$

$\Rightarrow f(\theta x + (1-\theta)y) = \theta f(x) + (1-\theta)f(y)$

---

(2) $\forall \underline{x} \in \mathbb{R}^n, \ \underline{b} \in \mathbb{R}^m, \ A \in \mathbb{R}^{m\times n}$

$f(\underline{x}) = \|A\underline{x} - \underline{b}\|^2 = (A\underline{x} - \underline{b})^T(A\underline{x} - \underline{b}) = \underline{x}^T A^T A\underline{x} - 2\underline{b}^T A\underline{x} + \underline{b}^T\underline{b}$

$\nabla f(\underline{x}) = 2A^T A\underline{x} - 2\underline{b}$, $\quad \nabla^2 f(\underline{x}) = 2A^T A = 2(\underline{a}_i^T \underline{a}_i) \ (i=1,\dots,n)$

$\nabla^2 f(\underline{x}) > 0 \Rightarrow f(\underline{x})$ is convex

Q3

$$H_{P,Q}(\mathcal{X}) = -\sum_{x_k \in \mathcal{X}} P(X=x_k) \cdot \log(Q(X=x_k))$$

$$H_P(\mathcal{X}) = -\sum_{x_k \in \mathcal{X}} p_k \cdot \log(p_k)$$

$$H_{P,Q}(\mathcal{X}) - H_P(\mathcal{X}) = \sum_{x_k \in \mathcal{X}} \left(p_k \cdot \log(p_k) - p_k \log(q_k)\right) = \sum_{x_k \in \mathcal{X}} p_k \log\left(\frac{p_k}{q_k}\right)$$

$$= -\sum_{x_k \in \mathcal{X}} p_k \log\left(\frac{q_k}{p_k}\right) = -E_P\left[\log\left(\frac{q_k}{p_k}\right)\right]$$

$(\log x)'' = -\frac{1}{x^2} < 0 \Rightarrow$ concave

$$E_P\left[\log\left(\frac{q_k}{p_k}\right)\right] \leq \log\left[\sum_{x_k \in \mathcal{X}} p_k \frac{q_k}{p_k}\right] = \log\left(\sum_{x_k \in \mathcal{X}} q_k\right)$$

$$= \log(1) = 0$$

Thus $H_{P,Q}(\mathcal{X}) - H_P(\mathcal{X}) \geq 0$

Since $p_k, q_k > 0$, when $-\sum_{x_k \in \mathcal{X}} p_k \log\left(\frac{q_k}{p_k}\right) = 0$

$$\log\left(\frac{q_k}{p_k}\right) = 0 \Rightarrow \frac{q_k}{p_k} = 1 \Rightarrow P = Q$$

Q.E.D.

$\underline{x}_i \in \mathbb{R}^d$, $y_i \in \mathbb{R}$, $i = 1, 2, \ldots, N$ $\qquad \underline{\bar{w}} = \hat{I}_d \underline{w} = [0, w_1, \ldots, w_d]^T$

$\uparrow$ b not included

Q4 (1) $\min\limits_{\underline{w}, b} \sum\limits_{i=1}^{N} (f_{\underline{w},b}(\underline{x}_i) - y_i)^2 + \lambda \underline{\bar{w}}^T \underline{\bar{w}}$

Write in the linear model, let $X = \begin{bmatrix} x_1^T \\ x_N^T \end{bmatrix} \in \mathbb{R}^{n \times d}$, $\underline{b} = \begin{bmatrix} b \\ b \\ \vdots \\ b \end{bmatrix} \in \mathbb{R}^d$

$\min\limits_{\underline{w}} (X\underline{w} + \underline{b} - \underline{y})^T (X\underline{w} + \underline{b} - \underline{y}) + \lambda \underline{\bar{w}}^T \underline{\bar{w}}$

Let the first derivative of the formula to be $\underline{0}$

$\frac{\partial}{\partial(\underline{w})} [(X\underline{w} + \underline{b} - \underline{y})^T (X\underline{w} + \underline{b} - \underline{y}) + \lambda (\hat{I}_d \underline{w})^T (\hat{I}_d \underline{w})] = \underline{0}$

$2X^T (X\underline{w} + \underline{b} - \underline{y}) + 2\lambda \hat{I}_d^T (\hat{I}_d \underline{w}) = \underline{0}$

$X^T (X\underline{w} + \underline{b} - \underline{y}) + \lambda \underline{w} = \underline{0} \Rightarrow (X^T X + \lambda I_d) \underline{w} = X^T (\underline{y} - \underline{b})$

$\frac{\partial}{\partial(\underline{b})} [(X\underline{w} + \underline{b} - \underline{y})^T (X\underline{w} + \underline{b} - \underline{y}) + \lambda (\hat{I}_d \underline{w})^T (\hat{I}_d \underline{w})] = \underline{0}$

$\underline{b}^T \underline{b} + 2\underline{b}^T X\underline{w} - 2\underline{b}^T \underline{y}$

$2\underline{b} + 2X\underline{w} - 2\underline{y} = \underline{0} \Rightarrow \underline{b} = \underline{y} - X\underline{w} \Rightarrow b = \frac{1}{N} \sum\limits_{i=1}^{N} (y_i - x_i^T \underline{w})$

Thus, $\underline{\hat{w}} = (X^T X + \lambda I_d)^{-1} X^T (\underline{y} - \underline{b})$

$b = \frac{1}{N} \sum\limits_{i=1}^{N} (y_i - \underline{x}_i^T \underline{w})$ is the closed-form solution

(2) Let $\underline{\bar{w}} = \operatorname{argmin}\limits_{\underline{\bar{w}}} J(\underline{\bar{w}})$

$J(\underline{\bar{w}}) = \frac{1}{2} \left[\sum\limits_{i=1}^{N} (f_{\underline{w},b}(\underline{x}_i) - y_i)^2 + \lambda \underline{\bar{w}}^T \underline{\bar{w}} \right]$

$= \frac{1}{2} [(X\underline{w} + \underline{b} - \underline{y})^2 + \lambda \underline{\bar{w}}^T \underline{\bar{w}}]$

The gradient descent is

$\underline{w} \leftarrow \underline{w} - \alpha [(X^T X + 2\lambda I_d) \underline{w} + X^T (\underline{y} - \underline{b})]$

$b \leftarrow b - \alpha \frac{1}{N} \sum\limits_{i=1}^{N} (y_i - \underline{x}_i^T \underline{w})$

Q5

We assume that $\underline{y} = W^T \phi(x) + e$; where $\underline{y}_i \in \mathbb{R}^2$, $x_i \in \{0, 1\}$, $\phi(0) = \begin{bmatrix}1\\0\end{bmatrix}$, $\phi(1) = \begin{bmatrix}0\\1\end{bmatrix}$
$e \sim N(0, \sigma^2)$

$$p(\underline{x} / W) \sim N(W^T \phi(x), \sigma^2)$$

$$\hat{W} = \underset{W}{\operatorname{argmax}} \log L(W; D) = \underset{W}{\operatorname{argmax}} \log \left(\prod_{i=1}^{6} p(W^T \phi(x), \sigma^2) \right)$$

$$= \underset{W}{\operatorname{argmax}} \sum_{i=1}^{6} \log(p(W^T\phi(x), \sigma^2))$$

$$= \underset{W}{\operatorname{argmax}} \sum_{i=1}^{6} \left[\log \frac{1}{\sqrt{2\pi\sigma^2}} - \frac{(\underline{y}_i - W^T\phi(x_i))^2}{2\sigma^2} \right]$$

$$= \underset{W}{\operatorname{argmax}} \left(-\log(8\pi^3\sigma^6) - \frac{1}{2\sigma^2} \sum_{i=1}^{6} (\underline{y}_i - W^T\phi(x_i))^2 \right)$$

$$= \underset{W}{\operatorname{argmax}} \left(-\frac{1}{2\sigma^2} \sum_{i=1}^{6} [\underline{y}_i - W^T\phi(x_i)]^2 \right)$$

$$= \underset{W}{\operatorname{argmin}} \sum_{i=1}^{6} (\underline{y}_i - W^T\phi(x_i))^T (\underline{y}_i - W^T\phi(x_i))$$

$$= \underset{W}{\operatorname{argmin}} \sum_{i=1}^{6} \left(\underline{y}_i^T \underline{y}_i - \underline{y}_i^T W^T \phi(x_i) - (W^T\phi(x_i))^T \underline{y}_i + (W^T\phi(x_i))^T W^T \phi(x_i) \right)$$

$$= \underset{W}{\operatorname{argmin}} \sum_{i=1}^{6} \left(\phi^T(x_i) W W^T \phi(x_i) - \underline{y}_i^T W^T \phi(x_i) - \phi^T(x_i) W \underline{y}_i \right)$$

$$= \underset{W}{\operatorname{argmin}} \left[\sum_{i=1}^{6} \phi^T(x_i) W W^T \phi(x_i) - \sum_{i=1}^{6} \underline{y}_i^T W^T \phi(x_i) - \sum_{i=1}^{6} \phi^T(x_i) W \underline{y}_i \right]$$

$$= \underset{W}{\operatorname{argmin}} \Big\{ 3 \times [1, 0] W W^T \begin{bmatrix}1\\0\end{bmatrix} + 3 \times [0, 1] W W^T \begin{bmatrix}0\\1\end{bmatrix}$$
$$- \left(\begin{bmatrix}-1\\-1\end{bmatrix} + \begin{bmatrix}-1\\-2\end{bmatrix} + \begin{bmatrix}-2\\-1\end{bmatrix} \right)^T W^T \begin{bmatrix}1\\0\end{bmatrix} - \left(\begin{bmatrix}1\\1\end{bmatrix} + \begin{bmatrix}1\\2\end{bmatrix} + \begin{bmatrix}2\\1\end{bmatrix} \right)^T W^T \begin{bmatrix}0\\1\end{bmatrix}$$
$$- [1, 0] W \left(\begin{bmatrix}-1\\-1\end{bmatrix} + \begin{bmatrix}-1\\-2\end{bmatrix} + \begin{bmatrix}-2\\-1\end{bmatrix} \right) - [0, 1] W \left(\begin{bmatrix}1\\1\end{bmatrix} + \begin{bmatrix}1\\2\end{bmatrix} + \begin{bmatrix}2\\1\end{bmatrix} \right) \Big\}$$

$$= \underset{W_{11}, W_{12}, W_{21}, W_{22}}{\operatorname{argmin}} \Big\{ 3(W_{11}^2 + W_{12}^2) + 3(W_{21}^2 + W_{22}^2) - [-4, -4] \begin{bmatrix}W_{11}\\W_{12}\end{bmatrix} - [4, 4] \begin{bmatrix}W_{21}\\W_{22}\end{bmatrix}$$
$$- [W_{11}, W_{12}] \begin{bmatrix}-4\\-4\end{bmatrix} - [W_{21}, W_{22}] \begin{bmatrix}4\\4\end{bmatrix} \Big\}$$

$$= \underset{W_{11} W_{12} W_{21} W_{22}}{\operatorname{argmin}} \Big[3(W_{11}^2 + W_{12}^2 + W_{21}^2 + W_{22}^2) + (4W_{11} + 4W_{12}) - (4W_{21} + 4W_{22}) + (4W_{11} + 4W_{12})$$
$$- 4(W_{21} + W_{22}) \Big]$$

$$= \underset{W_{11}, W_{12}, W_{21}, W_{22}}{\operatorname{argmin}} \left[3(W_{11}^2 + W_{12}^2 + W_{21}^2 + W_{22}^2) + 8W_{11} + 8W_{12} - 8W_{21} - 8W_{22} \right]$$

$$= \underset{W_{11} W_{12} W_{21} W_{22}}{\operatorname{argmin}} \left[(3W_{11}^2 + 8W_{11}) + (3W_{12}^2 + 8W_{12}) + (3W_{21}^2 - 8W_{21}) + (3W_{22}^2 - 8W_{22}) \right]$$

next page

Q5

$$\hat{W} = \begin{bmatrix} -\frac{4}{3} & -\frac{4}{3} \\ \frac{4}{3} & \frac{4}{3} \end{bmatrix}$$

The MLE for W is $\hat{W} = \begin{bmatrix} -\frac{4}{3} & -\frac{4}{3} \\ \frac{4}{3} & \frac{4}{3} \end{bmatrix}$

Report for DDA3020 Assignment 1 Coding Problems

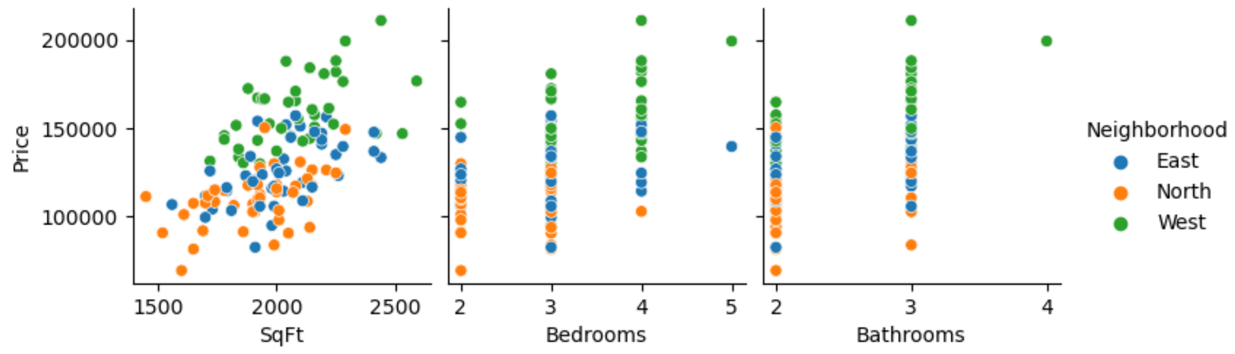
Problem 1:

Environment: *numpy*, *pandas*, *matplotlib*, *seaborn*, *sklearn*, *jupyter*

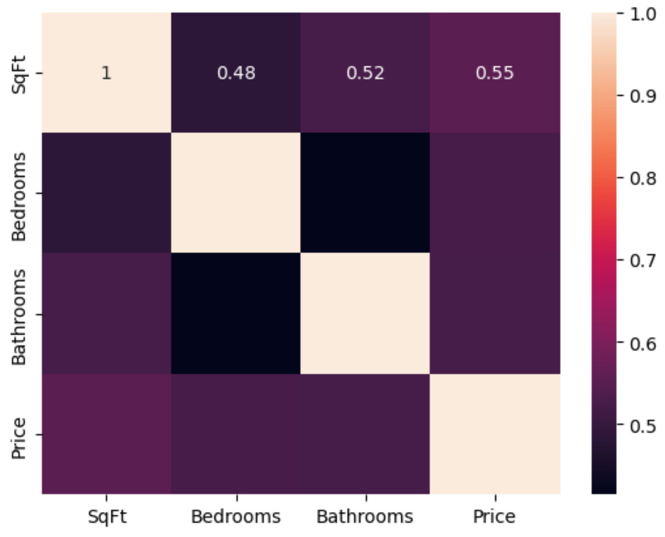
The task was to construct a linear regression model that predict the price of houses using 4 factors: the area of house, the number of bedrooms and bathrooms and the neighborhood, which means there are 4 independent variables "SqFt", "Bedrooms", "Bathrooms", "Neighborhood" and one dependent variable "Price".

The dataset is a csv file containing a table with 5 columns (5 variables listed above) and 128 rows (128 samples), with the "Neighborhood" being an object type and the others being integer type. In the first step, the dataset was loaded as *pandas.Dataframe*, briefly checked using the *Dataframe.info* and *Dataframe.describe* functions. Also noticed that there are only 3 values in the "Neighborhood" (The one-hot matrix would only contain 3 columns).

The second step was to visualize the dataset using *the seaborn* library.
"Price" against the 3 numeric attributes with data points colored differently based on the values of the "Neighborhood category using *seaborn.pairplot*:



Pairwise correlation on the data using *seaborn.heatmap*:



By generally analyzing the plots, there were some features:

1. House price was positively correlated with the area of the house, its bedroom number and bathroom number;
2. From prespective of neighborhood, houses in the North neighborhood tended to be cheaper while in the West neighborhood tended to be more expensive.

Based on the features I conjectured that houses in the North neighborhood were cheaper was related to less bedrooms and bathrooms on average while houses in the West neighborhood were opposite.

In the step 3, ColumnTransformer and OneHotEncoder functions from sklearn.compose and sklearn.preprocessing were used to convert the category column into an one-hot numeric matrix, which was left-fitted in to the X (128 samples * 4 variables) selected from the dataset to form a X_encode (128 samples * 6 variable, removing the "Neighborhood" and adding 3 dummy variables on the left carrying the same messages). What did 3 new lines in X_encode mean wasn't important here if we transform all test input data into the same form. Besides, I selected the variable "Price" to be y (128 samples * 1 variable). After that, split the data (containing one-to-one X_encode and y) into 80% of training and 20% of testing groups randomly using *train_test_split* function from *sklearn.model_selection* (For the consistency of the coding result in multiple code runnings, I randomly selected a certain random_state = 69).

In the final step, use the *LinearRegression* function in *sklearn.linear_model* to train the model with training part of X_encode and y. Then use the model trained and testing part of X_encode to predict the testing part of y. Calculate the RMSE for both training and testing parts:

Training RMSE: 14395.960374557913

Test RMSE: 15431.192673669071

Problem 2:

Environment: *numpy*, *matplotlib*, *sklearn*

The dataset was a diabetes in a csv file with 11 attributes: the first 10 baseline variables: "age", "sex", "bmi", "bp", and "s1" to "s6" contained information regarding age, sex, body mass index, average blood pressure and six blood serum measurements while the last "target" referred to a quantitative measure of disease progression one year after baseline, which is the target value predicted using the former 10 attributes.

At the very beginning, I defined 3 functions. "iter_times" and "step_sizes" were used to generate different iteration times and learning rates in the 10 training process while "rmse" was only for calculating RMSE and was not very necessary, only for short. There was also a list mapping the iteration times and step sizes, which was:

No.	1	2	3	4	5	6	7	8	9	10
Iter_times	1000	1000	1000	10000	50000	10000	1000	1000	10000	10000
Step_sizes	0.001	0.0007	0.0001	0.001	0.001	0.0001	a	b	a	b

a: (2-1/n)/2000 (decrease from 0.001 to 0.0005)

b: (2-1/n)/10000 (decrease from 0.0002 to 0.0001)

(1, 2, 3) & (4, 6) controlled iteration times to check the influence of learning rate;

(1, 4, 5) controlled learning rate to check the influence of iteration times;

(1, 7, 8) & (4, 9, 10) was set for checking the influence of decreasing learning rate;

(7, 9) & (8, 10) was set for checking the influence of iteration times under the same decreasing learning rate.

Here I made some hypotheses according to common knowledge:

> *H_{01}: The increase of iteration times would lead to a more accurate model (i.e. RMSE lower);*
>
> *H_{02}: The decrease of learning rate would lead to a more accurate model;*
>
> *H_{03}: A decreasing learning rate would lead to a more accurate model.*

Besides, use *load_diabetes* function in *sklearn.datasets* to load the data.

Step 1 and step 2 given by the problem were merged for consistency of the codes. I used *numpy* to form the input data matrix X, which was left-fit with a column of ones to be X_b, output data vector y, and a randomly generated initial w (i.e. the model), which was a 11-dimension vector (1 intercept on the left & 10 coefficients). The X_b and y were split into 80% for training and 20% for testing using the *train_test_split* from *sklearn.model_selection* (For the consistency of the coding result in multiple running times, I randomly selected a certain random_state = 69). The gradient descent model was conducted by calculating the negative first derivative of Jacobian of the model w with respect to w, which is the (X_b.T*(X_b*w)). Then multiply it with a step size generated by the step_size function and plus the result to the w in each iteration. The training RMSE at each iteration was also noted for plotting the loss curve. After training, both training RMSE and testing RMSE were reported and the loss curve of the training RMSE was plotted to check if there were cases of violating or non-convergence.

By repeating the above procedure for 10 times with different iteration times and step sizes shown above, I obtained the RMSE result (round 2):

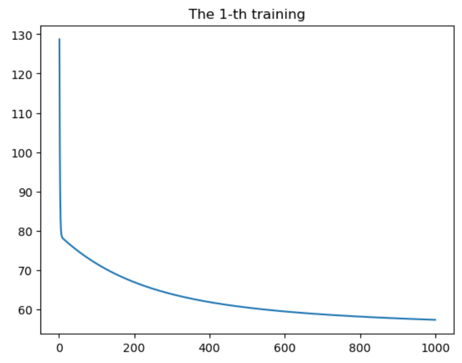
No.	1	2	3	4	5	6	7	8	9	10
Train	57.34	58.73	71.62	55.33	55.25	57.34	58.42	69.01	55.34	56.26
Test	49.31	50.33	61.15	47.12	47.08	49.31	50.12	58.64	47.11	48.29

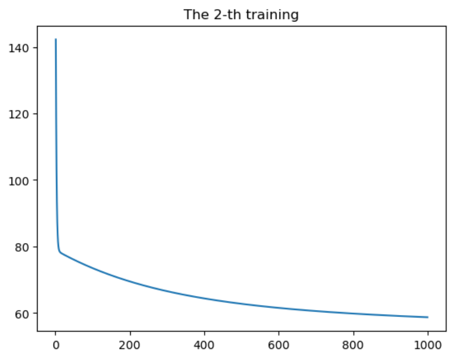
I briefly concluded some obvious features according to the former groups:

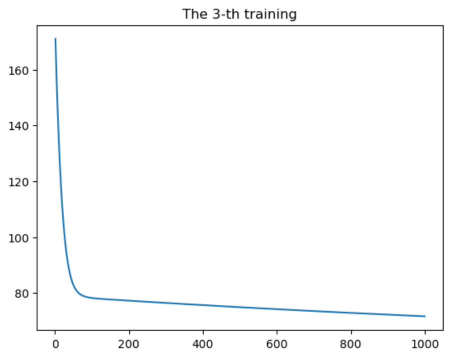
1. The RMSE's from testing samples were obviously bigger than the RMSE's from training samples, which was not intuitive since the model should fit the training data better;
2. For groups (1, 2, 3) & (4, 6), the RMSE didn't decrease with the decrease of learning rate as hypothesized;
3. For group (1, 4, 5), the RMSE decreased as hypothesized;
4. For groups (1, 7, 8) & (4, 9, 10), the last RMSE's in the 2 groups didn't decrease;
5. For groups (7, 8) & (9, 10), the RMSE seemed to increase with the decrease of learning rate (although it was changing).

I briefly analyzed the features above and made my conjectures below:

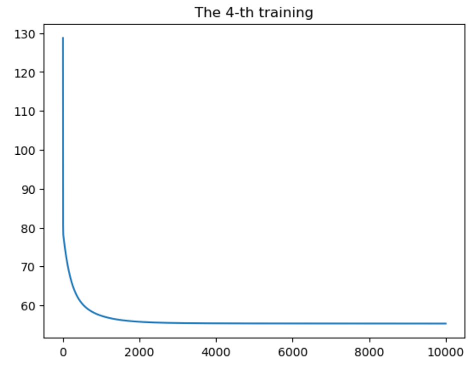
(1) For feature 1, the reason was probably data splitting. To be precise, data in the training set might have a higher variance while data in the testing set were generally more concentrated such that it fit the regression model better than data in the training set;
(2) By checking the loss function curves of the first 3 training processes, I could suggest that the models weren't fully trained due to few iteration times since the RMSE didn't seem to converge at the very end of the training processes, where the influence of magnitude is larger than accuracy. Therefore, a lower learning rate led to a worse trained model (i.e. larger RMSE).

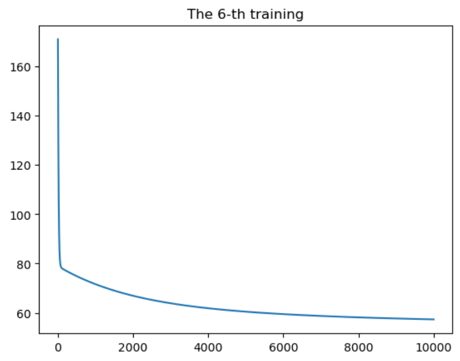






My conjecture on group (4, 6) were the same, that the 6th model didn't converge due to too small training rate, while the 4th model is fully trained and converge;





(3) For group (1, 4, 5), both 4th and 5th training obtained fully trained models, and the one with more iteration times received slightly lower RMSE as hypothesized;

(4) For groups (1, 7, 8) & (4, 9, 10), it was obvious that the 7th to the 10th cases weren't fully trained. Therefore, the lower the learning rate was, the worse the model was trained and the higher the RMSE. Because the mechanism of the feature was similar, I wouldn't show the pictures again. Besides, my conjecture for reason of feature 5 is the same.

It was a significant difficulty that I had no former experience that how many iteration times and what learning rate to set up as initial ones. The problem could be better solved with more precise iteration times and learning rates and could be further studied with the efficiency of different learning rates, especially decreasing learning rates.