Jingquan Li 11901014S DDA3020 Homework 1

1 Written Problems (50 pts.)

Problem 1 (10pts) Linear Algebra.

1. A rotation in 3D by angle α about the z axis is given by the following matrix:

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

Prove that $\mathbf{R}$ is an orthogonal matrix, i.e., $\mathbf{R}^T\mathbf{R} = \mathbf{I}$, for any α.

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

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

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

$$= \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} = I$$

∴ R is an orthogonal matrix for any α.

2. Prove that the eigenvalue of an orthogonal matrix must be 1 or -1.

proof: Let A be an orthogonal matrix. ∴ $A^TA = I$

Let λ be an eigenvalue of matrix A, and v be the corresponding eigenvector.

∴ $Av = \lambda v$ ∴ $\|Av\|^2 = \|\lambda v\|^2$ ∴ $\|Av\|^2 = \|\lambda\|^2\|v\|^2$

∵ $\|Av\|^2 = (Av)^T Av = v^T A^T A v = v^T (A^T A) v = v^T I v = v^T v = \|v\|^2$

∴ $\|v\|^2 = \|\lambda\|^2\|v\|^2$ ∵ v is the eigenvector ∴ $\|v\| \neq 0$

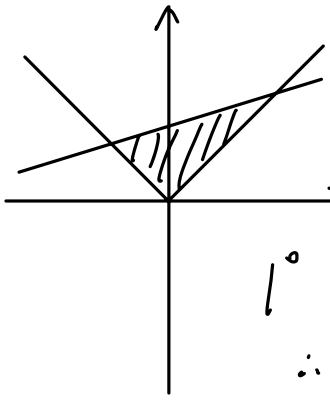
∴ $\|\lambda\|^2 = 1$ ∴ $\lambda = \pm 1$

∴ the eigenvalue of an orthogonal matrix must be 1 or -1.

Problem 2 (10pts) Optimization.
Prove that:

(1) $f(x) = |x|$ is convex; $x \in R$

Proof: For $x_1, x_2 \in R$, let $x_1 < x_2$



Let the equation of the line segment connecting the two points $(x_1, |x_1|)$, $(x_2, |x_2|)$ be:

$g(x) = kx + b \quad (k \neq 0) \qquad \therefore k = \frac{|x_2| - |x_1|}{x_2 - x_1}$

1° $x_2 \geq 0$, $x_1 \geq 0$, $x_2 > x_1$

$\therefore k = \frac{x_2 - x_1}{x_2 - x_1} = 1$ plug in $(x_1, |x_1|) \Rightarrow |x_1| = x_1 + b$

$\Rightarrow x_1 = x_1 + b \Rightarrow b = 0 \qquad \therefore g(x) = x$

Let $x \in (x_1, x_2) \qquad \therefore g(x) - f(x) = x - |x| = x - x = 0$

2° $x_2 \geq 0$, $x_1 < 0$, $x_2 > x_1$

$\therefore k = \frac{x_2 + x_1}{x_2 - x_1}$ plug in $(x_1, |x_1|) \Rightarrow |x_1| = \frac{x_2 + x_1}{x_2 - x_1} x_1 + b$

$\Rightarrow -x_1 = \frac{x_2 + x_1}{x_2 - x_1} x_1 + b \qquad \therefore b = -x_1 - \frac{x_2 + x_1}{x_2 - x_1} x_1$

$= -\left(1 + \frac{x_2 + x_1}{x_2 - x_1}\right) x_1 = -\frac{2x_2}{x_2 - x_1} x_1 = -\frac{2x_1 x_2}{x_2 - x_1}$

$\therefore g(x) = \frac{x_2 + x_1}{x_2 - x_1} x - \frac{2x_1 x_2}{x_2 - x_1}$

Let $x \in (x_1, x_2)$

① $x \geq 0 \qquad \therefore g(x) - f(x) = \frac{x_2 + x_1}{x_2 - x_1} x - \frac{2x_1 x_2}{x_2 - x_1} - x = \frac{2x_1}{x_2 - x_1} x - \frac{2x_1 x_2}{x_2 - x_1}$

$= \frac{2x_1}{x_2 - x_1}(x - x_2) \qquad \because 2x_1 < 0, x_2 - x_1 > 0, x - x_2 < 0$

$\therefore g(x) - f(x) > 0$

② $x < 0 \qquad \therefore g(x) - f(x) = \frac{x_2 + x_1}{x_2 - x_1} x - \frac{2x_1 x_2}{x_2 - x_1} - (-x) = \frac{2x_2}{x_2 - x_1} x - \frac{2x_1 x_2}{x_2 - x_1}$

$= \frac{2x_2}{x_2 - x_1}(x - x_1) \qquad \because 2x_2 \geq 0, x_2 - x_1 > 0, x - x_1 > 0$

$\therefore g(x) - f(x) \geq 0$

3° $x_2 < 0$, $x_1 < 0$, $x_2 > x_1$

$\therefore k = \frac{-x_2 + x_1}{x_2 - x_1} = -1$ plug in $(x_1, |x_1|) \Rightarrow |x_1| = -x_1 + b$

$\therefore -x_1 = -x_1 + b \quad \therefore b = 0 \quad \therefore g(x) = -x$

Let $x \in (x_1, x_2) \quad \therefore g(x) - f(x) = -x - |x| = -x - (-x) = -x + x = 0$

In conclusion, $g(x) - f(x) \geq 0$.

$\therefore$ By definition of convex function, $f(x) = |x|$ is convex.

(2) $f(\mathbf{x}) = \|\mathbf{Ax} - \mathbf{b}\|^2$ is convex, where $\mathbf{A}$ is a matrix.

Proof: $f(x) = \|Ax-b\|^2 = (Ax-b)^T(Ax-b) = (x^TA^T - b^T)(Ax-b)$

$= A^TAx^2 - A^Tbx^T - Ab^Tx + bb^T$

$\therefore \nabla f(x) = 2A^TAx - A^Tb - Ab^T$

$\therefore \nabla^2 f(x) = 2A^TA \quad \because A^TA$ is positive semidefinite

$\therefore f(x) = \|Ax-b\|^2$ is convex.

Problem 3 (10pts) Information Theory.
Proof that cross-entropy is not smaller than entropy, i.e., $H_{P,Q}(\mathcal{X}) \geq H_P(\mathcal{X})$, and the equality holds only when $P = Q$.

Proof: cross-entropy: $H_{P,Q}(X) = -\sum_{x_k \in X} P(X = x_k) \cdot \log(Q(X = x_k))$

entropy: $H_P(X) = -\sum_{x_k \in X} P(X = x_k) \log(P(X = x_k))$

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

$= \sum_{x_k \in X} P(X = x_k) \log \frac{P(X = x_k)}{Q(X = x_k)}$

$= D_{P,Q}(X)$

$\because D_{P,Q}(X) \geq 0$ and $D_{P,Q}(X) = 0$ when $P = Q$

$\therefore H_{P,Q}(X) - H_P(X) \geq 0 \Rightarrow H_{P,Q}(X) \geq H_P(X)$, and the equality holds only when $P = Q$

Problem 4 (10pts) Linear Regression.
Suppose we have training data $\{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_N, y_N)\}$, where $x_i \in \mathbb{R}^d$ and $y_i \in \mathbb{R}, i = 1, 2, \ldots, N$. Consider $f_{\mathbf{w},b}(\mathbf{x}_i) = \mathbf{x}_i^T\mathbf{w} + b$, where $\mathbf{w} = [w_1, w_2, \ldots, w_d]^T$.

(1) Find the closed-form solution of the following problem

$$\min_{\mathbf{w},b} \sum_{i=1}^{N} (f_{\mathbf{w},b}(\mathbf{x}_i) - y_i)^2 + \lambda \bar{\mathbf{w}}^T \bar{\mathbf{w}},$$

where $\bar{\mathbf{w}} = \hat{\mathbf{I}}_d \mathbf{w} = [0, w_1, w_2, \ldots, w_d]^T$. Note that $\hat{\mathbf{I}}_d = \begin{bmatrix} 0 & 0 & 0 & \ldots & 0 \\ 1 & 0 & 0 & \ldots & 0 \\ 0 & 1 & 0 & \ldots & 0 \\ & & & \vdots & \\ 0 & \ldots & 0 & 0 & 1 \end{bmatrix} \in \mathbf{R}^{(d+1)\times d}$

Solution: $\min_{w,b} \sum_{i=1}^{N} (f_{w,b}(x_i) - y_i)^2 + \lambda \bar{w}^T \bar{w}$

$\Rightarrow \min_{w,b} (Xw-y)^T(Xw-y) + \lambda \bar{w}^T w$

$\Rightarrow \frac{\partial}{\partial w} (Xw-y)^T(Xw-y) + \lambda \bar{w}^T w = 0$

$\Rightarrow \frac{\partial}{\partial w} (w^T X^T - y^T)(Xw-y) + \lambda w^T \hat{I}_d^T \hat{I}_d w = 0$

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

$\Rightarrow 2X^T X w - 2X^T y + 2\lambda \hat{I}_d w = 0$

$\Rightarrow X^T X w + \lambda \hat{I}_d w = X^T y$

$\Rightarrow (X^T X + \lambda \hat{I}_d) w = X^T y$

$\Rightarrow w = (X^T X + \lambda \hat{I}_d)^{-1} X^T y$

($X^T X + \lambda \hat{I}_d$ is invertible, given that $\lambda > 0$)

(2) Show how to use gradient descent to solve the problem.

Solution: cost function: $J(w) = \frac{1}{2} \sum_{i=1}^{N} (x_i^T w + b - y_i)^2 + \lambda \bar{w}^T w$

$= \frac{1}{2} (X\bar{w} - y)^2 + \lambda w^T \hat{I}_d w$

$\therefore \bar{w}^* = \arg_{\bar{w}} \min J(\bar{w})$

$\therefore w \leftarrow w - \alpha \frac{\partial J(w)}{\partial w}$

$= w - \alpha X^T (Xw - y) - 2\lambda \hat{I}_d w$

Repeat this updating step until $\frac{\partial J(w)}{\partial w}$ converges to zero.

Problem 5 (10pts) MLE.

Consider a linear regression model with a 2 dimensional response vector $\boldsymbol{y}_i \in \mathbb{R}^2$. Suppose we have some binary input data, $x_i \in \{0, 1\}$. The training data is as follows:

x	y
0	$(-1, -1)^T$
0	$(-1, -2)^T$
0	$(-2, -1)^T$
1	$(1, 1)^T$
1	$(1, 2)^T$
1	$(2, 1)^T$

Let us embed each x_i into 2 d using the following basis function:

$$\phi(0) = (1, 0)^T, \quad \phi(1) = (0, 1)^T$$

The model becomes

$$\hat{\boldsymbol{y}} = \mathbf{W}^T \phi(x)$$

where $\mathbf{W}$ is a 2×2 matrix. Compute the MLE for $\mathbf{W}$ from the above data.

Solution: $L(w) = \prod_{i=1}^{b} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2\sigma^2}(y_i - w^T\varphi(x_i))^2\right)$

$\log L(w) = -3\log(2\pi\sigma^2) - \frac{1}{2\sigma^2}\sum_{i=1}^{b}(y_i - w^T(\varphi(x_i))^2$

$\frac{\partial}{\partial w}\log L(w) = -\frac{1}{2\sigma^2}\sum_{i=1}^{b}\frac{\partial}{\partial w}(y_i - w^T\varphi(x_i))^T(y_i - w^T\varphi(x_i))$

$= -\frac{1}{2\sigma^2}\sum_{i=1}^{b}\frac{\partial}{\partial w}(y_i^T - \varphi(x_i)^T w)(y_i - w^T\varphi(x_i))$

$= -\frac{1}{2\sigma^2}\sum_{i=1}^{b}\frac{\partial}{\partial w}(y_i^T y_i - y_i^T w^T\varphi(x_i) - \varphi(x_i)^T w y_i + \varphi(x_i)^T w\, w^T\varphi(x_i))$

$= -\frac{1}{2\sigma^2}\sum_{i=1}^{b}\frac{\partial}{\partial w}(y_i^T y_i - 2\varphi(x_i)^T w y_i + \varphi(x_i)^T w w^T\varphi(x_i))$

$= -\frac{1}{2\sigma^2}\sum_{i=1}^{b}(-2y_i\varphi(x_i)^T + 2\varphi(x_i)\varphi(x_i)^T w)$

To maximum the likelihood function, $\frac{\partial}{\partial w}\log L(w) = 0$

$\therefore \sum_{i=1}^{b}(-y_i\varphi(x_i)^T + \varphi(x_i)\varphi(x_i)^T w) = 0$

$\therefore \sum_{i=1}^{b}\varphi(x_i)\varphi(x_i)^T w = \sum_{i=1}^{b} y_i\varphi(x_i)^T$

$\therefore w = \left[\sum_{i=1}^{b}\varphi(x_i)\varphi(x_i)^T\right]^{-1}\left(\sum_{i=1}^{b} y_i\varphi(x_i)^T\right)$

$\because \sum_{i=1}^{b}\varphi(x_i)\varphi(x_i)^T = 3\begin{pmatrix}1\\0\end{pmatrix}\begin{pmatrix}1 & 0\end{pmatrix} + 3\begin{pmatrix}0\\1\end{pmatrix}\begin{pmatrix}0 & 1\end{pmatrix}$

$= 3\times\begin{pmatrix}1 & 0\\0 & 0\end{pmatrix} + 3\begin{pmatrix}0 & 0\\0 & 1\end{pmatrix} = \begin{pmatrix}3 & 0\\0 & 3\end{pmatrix}$

$\sum_{i=1}^{b} y_i\varphi(x_i)^T = \begin{pmatrix}-1\\-1\end{pmatrix}\begin{pmatrix}1 & 0\end{pmatrix} + \begin{pmatrix}-1\\-2\end{pmatrix}\begin{pmatrix}1 & 0\end{pmatrix} + \begin{pmatrix}-2\\-1\end{pmatrix}\begin{pmatrix}1 & 0\end{pmatrix} + \begin{pmatrix}1\\1\end{pmatrix}\begin{pmatrix}0 & 1\end{pmatrix} + \begin{pmatrix}1\\2\end{pmatrix}\begin{pmatrix}0 & 1\end{pmatrix}$

$+ \begin{pmatrix}2\\1\end{pmatrix}\begin{pmatrix}0 & 1\end{pmatrix}$

$= \begin{pmatrix}-1 & 0\\-1 & 0\end{pmatrix} + \begin{pmatrix}-1 & 0\\-2 & 0\end{pmatrix} + \begin{pmatrix}-2 & 0\\-1 & 0\end{pmatrix} + \begin{pmatrix}0 & 1\\0 & 1\end{pmatrix} + \begin{pmatrix}0 & 1\\0 & 2\end{pmatrix} + \begin{pmatrix}0 & 2\\0 & 1\end{pmatrix}$

$= \begin{pmatrix}-4 & 4\\-4 & 4\end{pmatrix}$

$\therefore w = \begin{pmatrix}3 & 0\\0 & 3\end{pmatrix}^{-1}\begin{pmatrix}-4 & 4\\-4 & 4\end{pmatrix} = \frac{1}{9}\begin{pmatrix}3 & 0\\0 & 3\end{pmatrix}\begin{pmatrix}-4 & 4\\-4 & 4\end{pmatrix} = \begin{pmatrix}\frac{1}{3} & 0\\0 & \frac{1}{3}\end{pmatrix}\begin{pmatrix}-4 & 4\\-4 & 4\end{pmatrix} = \begin{pmatrix}-\frac{4}{3} & \frac{4}{3}\\-\frac{4}{3} & \frac{4}{3}\end{pmatrix}$

Coding Problems Report

Problem 1

1. Task Description

The task involves regression analysis on a dataset named "house prices.csv", containing 128 samples with 5 features: SqFt, Bedrooms, Bathrooms, Neighborhood, and Price. The objective is to predict house prices using appropriate attributes.

2. Dataset Description

The dataset contains 128 entries with 5 columns: SqFt, Bedrooms, Bathrooms, Neighborhood, and Price. There are no missing values in any column. SqFt ranges from 1450 to 2590 square feet, with a mean of approximately 2000 square feet. The number of Bedrooms ranges from 2 to 5, with a mean of approximately 3. The number of Bathrooms ranges from 2 to 4, with a mean of approximately 2.45. The Price of houses ranges from $69,100 to $211,200, with a mean of approximately $130,427.

3. Linear Model

(1) Linear Hypothesis Function

Here I utilize thelinear regression model from the sklearn library to train and evaluate the dataset. The linear regression model is initialized using the LinearRegression class from sklearn.linear_model.

We can build a linear model $f_{w,b}(x)$, i.e., linear hypothesis function,

$$f_{w,b}(x) = x^T w + b$$

where w is a d-dimensional vector of parameters, and the bias parameter b is a real number.

(2) Root Mean Squared Error (RMSE)

RMSE is a measure of the differences between values predicted by a model and the observed values. It is calculated as the square root of the mean of the squared differences between predicted and observed values. The formula is:

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \bar{y}_i)^2}$$

4. Implement steps and important outputs

Step 1: Load data

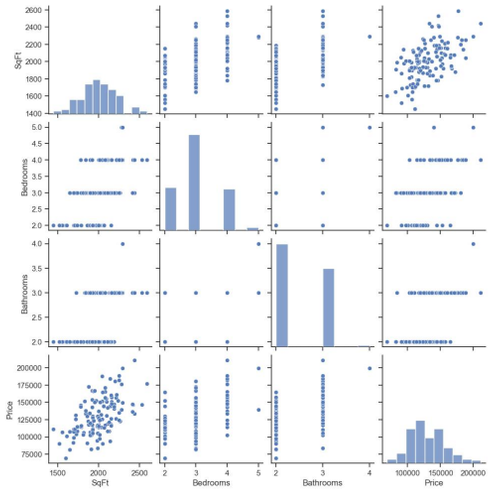
Load the csv file into pandas.Dataframe using pandas library. Convert the data type of the “Neighborhood” attribute to “category” type using Dataframe.astype function. Use Dataframe.info

and Dataframe.describe functions to check the dataset.

	SqFt	Bedrooms	Bathrooms	Price
count	128.000000	128.000000	128.000000	128.000000
mean	2000.937500	3.023438	2.445312	130427.343750
std	211.572431	0.725951	0.514492	26868.770371
min	1450.000000	2.000000	2.000000	69100.000000
25%	1880.000000	3.000000	2.000000	111325.000000
50%	2000.000000	3.000000	2.000000	125950.000000
75%	2140.000000	3.000000	3.000000	148250.000000
max	2590.000000	5.000000	4.000000	211200.000000

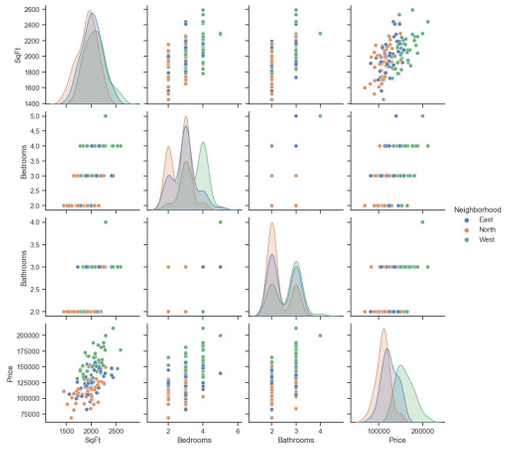
Step 2: Visualize data

(1) Use seaborn.pairplot function to plot the "Price" against each numeric attributes "SqFt", "Bedrooms" and "Bathrooms" with data points colored differently based on the values of the "Neighborhood" category attributes.



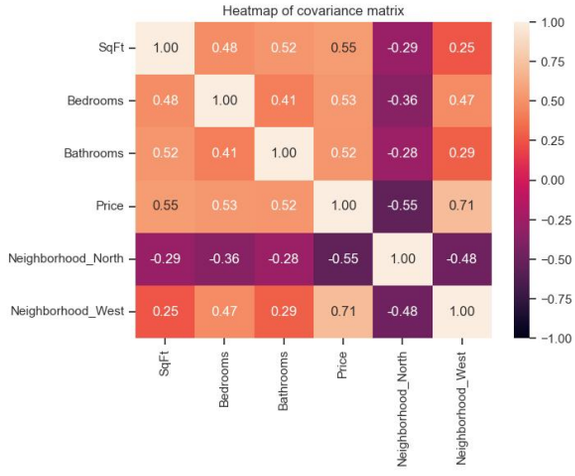
The figures clearly indicate the relationship between different attributes. From the output figures, I think the "Price" is relevantly highly correlated with "SqFt", since the dots in the figure SqFt vs. Price are grouped around a line.

(2) Data visualization with category attribute "Neighborhood" using pairlot.



From the last row of the figures above, I think price is highly correlated with the neighborhood where the house is located. As is shown from the figurea, houses in West neighborhood usually have higher price, while houses in North neighborhood usually have lower price. And the prices of houses in East neigborhood are usually between the above two.

(3) Plot pairwise correlation using heatmap



***Note**: When I was plotting the heatmap for the first time, I found there are only data in the first row of the map. Therefore, I searched online for solutions. It states that the problem is caused by version contradiction. Finally, I tried to downgrade the version of matplotlib from 3.8.0 to 3.7.3, then I restart VScode. It works and datas are shown in every cell of the heatmap.*

The intensity of colors in the heatmap indicates the strength of correlation between the attributes. Darker colors typically represent stronger correlations, while lighter colors indicate weaker correlations. In this figure, I just set 0.55 as the threshold value. It is clearly inferred that “SqFt” and “Neighborhood” is strongly correlated with “price”.

Step 3: Process the category variable and split the dataset

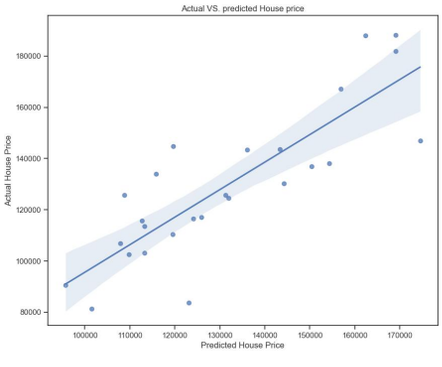
Use sklearn library to process the category variable. Use ColumnT ransformer function in sklearn.compose and OneHotEncoder function in sklearn.preprocessing to convert the category column into a one-hot numeric matrix in the dataset. Use sklearn library to split data into train and test subset.

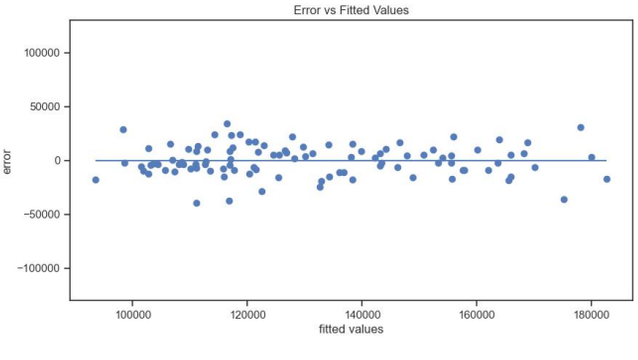
Step 4: Train and evaluate a linear regression model

Fit the model using training data and predict on training and testing datasets. Calculate the RMSE for both datasets.

The training RMSE is 14180.603255747075, and the testing RMSE is 15766.97214688351.

Plot the figure of “actual values vs. predicted values” and “error vs. fitted values”.





From the figures above, we can see that the predicted house prices are quite near the actual house prices. And the errors are floating around the zero line, which indicates the perfectly fitted line. Therefore, we can conclude that the performance of our prediction is quite good, and our selection of correlated attributes is awesome, too.

Problem 2

1. Task Description

The problem involves implementing linear regression from scratch using numpy, evaluating the model's performance using RMSE, and analyzing the effects of various parameters on model performance through repeated trials.

2. Dataset Description

The diabetes dataset in the sklearn package comprises 442 samples from diabetic patients, each containing 11 attributes. The first 10 attributes, including age, sex, body mass index, average blood pressure, and six blood serum measurements, have been preprocessed by mean-centering and scaling. The final attribute quantifies disease progression one year post-baseline. The objective is to utilize the baseline variables (attributes 1-10) to predict the progression of the disease (attribute 11).

3. Linear Model

(1) Cost function

The cost function (or loss function) of a linear regression model typically uses the Mean Squared Error (MSE) to measure the difference between the model's predicted values and the true values.

The formula is as follows:

$$J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^i) - y^i)^2$$

Where m is the number of data, $h_\theta(x^i)$ is the predicted value, y^i is the actual value.

In this formula, we calculate the square of the difference between the predicted values and the true values for each sample, sum all the squared differences, and then take the average. The final cost function is half of this mean squared error. The objective is to minimize the cost function to make the model's predicted values as close to the true values as possible.

(2) Gradient Descent Algorithm

The basic idea behind gradient descent is to update the parameters of the model in the direction that reduces the cost function the most. This direction is determined by the gradient of the cost function with respect to each parameter. The steps are as follows:

- Initialize the parameters of the model randomly or with some initial values.

- Compute the gradient of the cost function with respect to each parameter.

- Update each parameter by subtracting a fraction of the gradient from its current value. This fraction is called the learning rate, and it determines the size of the steps we take in the parameter space.

- Repeat steps 2 and 3 until the change in the cost function becomes negligible or a predefined number of iterations is reached.

The formula for updating the parameters of the model in each iteration of gradient descent is as follows:

$$\mathbf{w} \leftarrow \mathbf{w} - \alpha \frac{\partial J(\mathbf{w})}{\partial \mathbf{w}}, \quad \frac{\partial J(\mathbf{w})}{\partial \mathbf{w}} = \mathbf{X}^\top (\mathbf{X}\mathbf{w} - \mathbf{y})$$

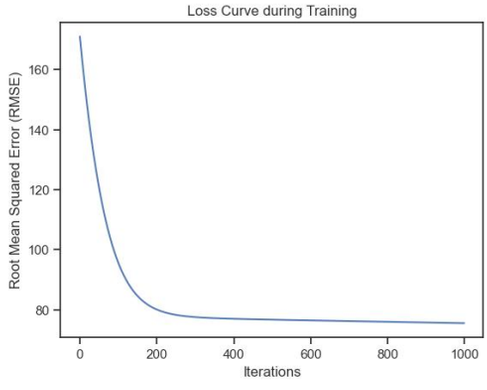
4. Implement steps and important outputs

Step 1: Load the dataset and construct the linear regression model

Load the diabetes dataset. Define the cost function and gradient descent function to train the linear regression model.

Step 2: Split the dataset

Split the data into training and testing sets. Train a linear regression model. Make prediction on the testing data and calculate the RMSE. Plot the loss curve during training. The Training Error (in RMSE) is 75.5149134242802, and the Testing Error (in RMSE) is 70.78385118285611. The loss curve during training is as follows:



Step 3: Repeat the trial steps

Repeat the splitting, training, and testing for 10 times with different parameters such as step size, iterations. Compute RMSE for training and testing data for every trial.

Trial	Learning Rate	Iteration number	Training RMSE	Testing RMSE
1	0.08768050829387422	901	64.97623545988635	62.75010091514034
2	0.03521759565955094	556	72.75642453223223	72.8021191277499
3	0.04053492421038708	923	69.70970562619212	69.97123311745507
4	0.06925355458074002	486	69.76558145002915	72.41291040329193
5	0.013673862446637477	692	75.56802167700405	71.94738769858077

6	0.011777856130416436	804	75.09544437508602	73.74636792951897
7	0.041978235680492686	264	73.83993114498472	76.56029805784054
8	0.02471675276038744	529	74.81244987357903	70.33574123586904
9	0.010265142033388045	259	76.82559348655353	79.41822538701946
10	0.03334253482154185	805	72.30158358114656	66.46128530940095

(1) Learning Rate: Trial 1 and Trial 5 have the lowest training and testing errors, suggesting that the learning rate used in these trials (0.08768 and 0.01367, respectively) may have been suitable for the given dataset. Trials with higher or lower learning rates generally resulted in higher testing errors, indicating that they might have overshot the minimum or converged too slowly.

(2) Iteration Number: There is no clear trend in how the iteration number influences the RMSE based on the output data. However, it's worth noting that Trial 1, with the highest iteration number (901), achieved relatively lower training and testing errors compared to other trials with similar learning rates. This suggests that more iterations may have helped the model converge to a better solution in this case.

Overall, the choice of learning rate appears to have a more significant impact on RMSE compared to the iteration number in the context of the provided data. It's essential to strike a balance between these parameters to achieve optimal performance in training machine learning models. Additionally, further experimentation and tuning may be necessary to find the best combination of parameters for the dataset.