NYCU Introduction to Machine Learning, Homework 1

[111550196], [Jorge Tyrakowski 狄豪飛]

The screenshot and the figures we provided below are just examples. **The results below are not guaranteed to be correct.** Please make sure your answers are clear and readable, or no points will be given.

Please also remember to convert it to a pdf file before submission.

You should use English to answer the questions (-5pt if not report in English).

After reading this paragraph, you should delete this paragraph.

Part. 1, Coding (60%):

(10%) Linear Regression Model - Closed-form Solution

1. (10%) Show the weights and intercepts of your linear model.

```
(ML-NYCU) (base) dihaofei@dihaofei:~/NYCU/Machine Learning/HW1$ /home/dihaofei/conda-envs/ML-NYCU/bin/python "/home/dihaofei/NYCU/Machine Learning/HW1/main.py"
2024-09-30 21:48:49.709 | INFO     | __main__:main:94 - LR_CF.weights=array([2.8491883 , 1.0188675 , 0.48562739, 0.1937254 ]), LR_CF.intercept=-33.8223
```

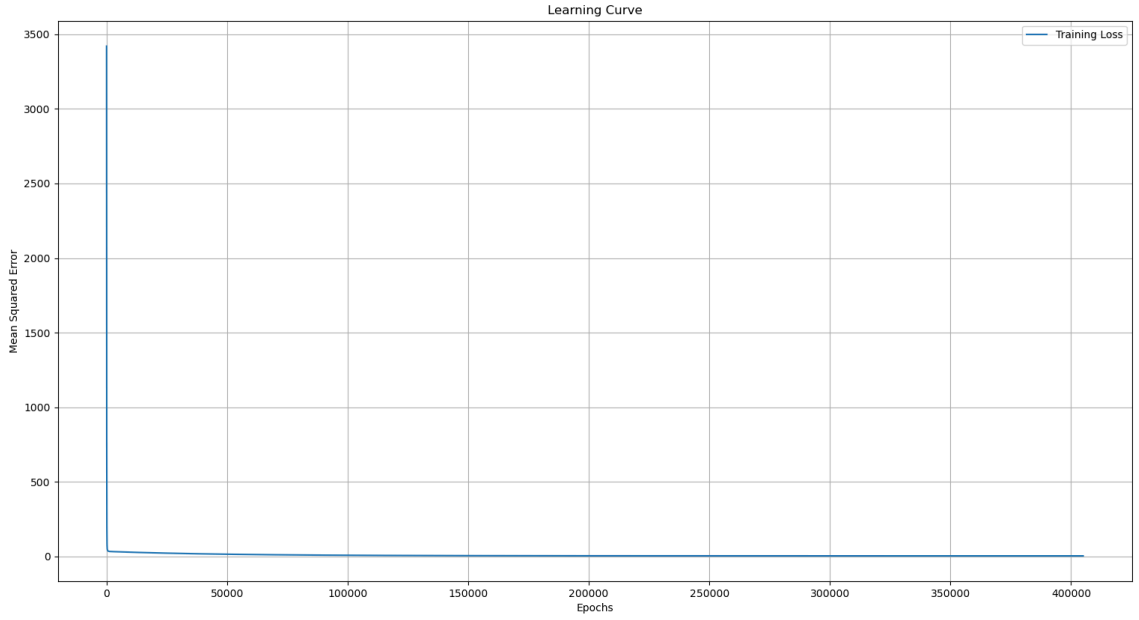
(40%) Linear Regression Model - Gradient Descent Solution

2. (10%)
   - Show the hyperparameters of your setting (e.g., learning rate, number of epochs, batch size, etc.).
   - Show the weights and intercepts of your linear model.

```
LR_GD.fit(train_x, train_y, learning_rate=0.000382, epochs=405000)
```

```
2024-09-30 21:54:53.276 | INFO     | __main__:main:99 - LR_GD.weights=array([2.83793383, 1.01533535, 0.45041785, 0.18565711]),
LR_GD.intercept=-33.2379
```

3. (10%) Plot the learning curve. (x-axis=epoch, y-axis=training loss)

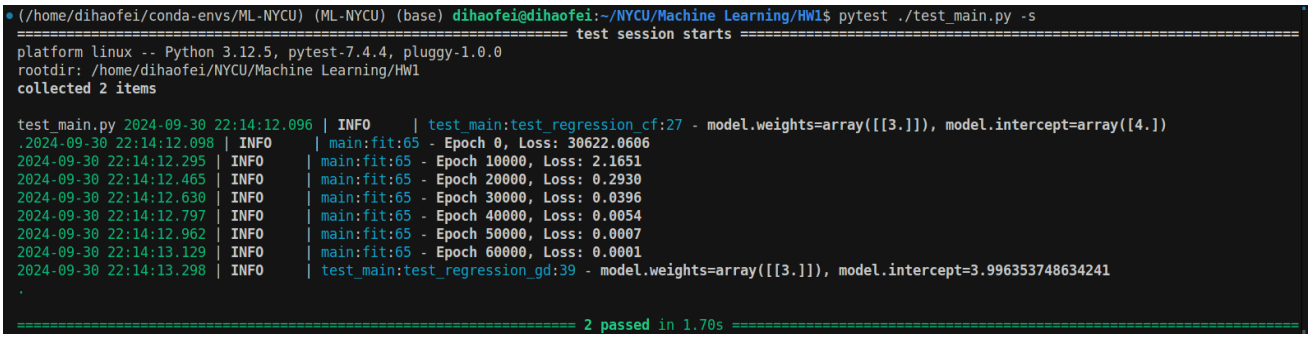


4. (20%) Show your MSE.cf, MSE.gd, and error rate between your closed-form solution and the gradient descent solution.

```
2024-09-30 21:58:29.028 | INFO     | __main__:main:108 - Mean prediction difference: 0.0780
2024-09-30 21:58:29.029 | INFO     | __main__:main:113 - mse_cf=4.1997, mse_gd=4.1996. Difference: 0.004%
```

(10%) Code Check and Verification

5. (10%) Lint the code and show the PyTest results.



Part. 2, Questions (40%):

1. **(10%) How does the presence of outliers affect the performance of a linear regression model? How should outliers be handled? List at least two methods.**

The presence of **outliers** (data points that stray significantly from the overall pattern of a dataset) can really impact how well a **linear regression model** performs. Let's dive into the different ways outliers affect the model and how we can handle them effectively.

First, outliers can cause **bias in the coefficient estimates**. Linear regression works by finding the best-fitting line that minimizes the sum of squared residuals, which are the differences between the observed and predicted values. When there are outliers with extreme values in either the dependent variable (Y) or the independent variables (X), they can heavily influence this process. This means that the estimated coefficients might not accurately represent the true relationship between the variables. For example, if most data points follow a clear linear trend but a few points are way above or below this trend, the regression line might tilt toward these outliers, making the slope less reflective of the majority of the data.

Another issue is that outliers can **increase the variance in model predictions**. Since outliers can skew the regression line, the model's predictions become more variable. This makes the model less stable when applied to different datasets because it's too sensitive to a small number of extreme data points. Higher variance means the model is less reliable and its ability to generalize to new data is reduced.

Outliers also **violate key assumptions of linear regression**. Linear regression relies on several important assumptions, such as linearity, homoscedasticity (constant variance of errors), and normally distributed residuals. Outliers can disrupt these assumptions. For instance, they can cause **heteroscedasticity**, where the variance of the residuals increases with the magnitude of the predicted values.

Another significant problem is the **distortion of the regression line**. Linear regression uses the Ordinary Least Squares (OLS) method to minimize the squared residuals. Since outliers tend to have large residuals because they don't fit the general trend, the OLS method gives more weight to these larger errors. This means that outliers can pull the regression line toward themselves, resulting in a line that doesn't accurately reflect the true relationship between the independent and dependent variables.

To handle outliers in linear regression, we need to ensure that these unusual data points don't negatively impact our model. One common strategy is to **remove outliers after identifying them**. This is sensible if the outliers result from errors in data collection or measurement. For instance, if we find extreme values that don't seem to represent the population we're studying, excluding them might be the best option. To detect outliers, we can use **visual tools** and **statistical methods** to help identify outliers. However, we must be cautious when removing outliers because this can lead to a loss of valuable information. It's crucial to ensure that these points are genuinely errors and not part of the natural variability in our data.

Another method is to **transform the data**. This approach is useful when outliers are not due to errors but are inherent in the data distribution. By applying transformations, we can reduce the influence of these extreme values. For example, a **logarithmic transformation** can compress the range of our data, which is particularly helpful if our data shows exponential growth or has a long-tailed distribution. Similarly, a **square root transformation** can be effective for moderately skewed data with positive outliers. Another technique called **Winsorizing** involves capping the outliers at a certain percentile, such as the 5th or 95th percentile. This way, we limit the impact of extreme values without completely removing them from our dataset. **Imputation of Outliers** It also can be a method, instead of removing outliers, replace them with more representative values, such as the mean, median, or values predicted by a model. This approach maintains the dataset's size while mitigating the influence of extreme values.

2. **(15%) How do different values of learning rate (too large, too small…) affect the convergence of optimization? Please explain in detail.**

The **learning rate** controls how big of a step the model takes when trying to minimize the error. If the learning rate is too large or too small, it can drastically affect how well the model learns, or if it even learns at all.

If the learning rate is set too high, the model takes large steps in each iteration. This can be a big problem because instead of gradually moving towards the point where the error is smallest (the **global minimum**), the model might overshoot and miss it completely. It could even keep bouncing back and forth, always overshooting and never really getting close to the minimum. In some cases, a large learning rate can cause the model to completely "diverge," meaning it doesn't just miss the minimum but keeps going in the wrong direction, making the error larger over time. This makes training very unstable, and you might see the loss (or error) jumping around a lot instead of steadily going down.

On the other hand, if the learning rate is too small, the model will take tiny steps in each iteration. This makes the training process really slow because it takes forever to make noticeable progress towards the minimum. Even after a large number of iterations, the model might still be far from the optimal solution. Another issue with a small learning rate is that the model can get stuck in **local minima** or flat areas of the error surface, where it doesn't make much progress. When this happens, it might never escape those spots and find a better solution. Plus, because the steps are so small, it requires a lot more computation and time to complete the training, which isn't ideal for complex models or large datasets.

When we talk about **convergence**, we're referring to how well the model approaches the optimal point, where the loss is minimized. If the learning rate is too high, the model may never converge because it's taking such big steps that it keeps missing the minimum. If it's too low, the model might converge too slowly or even get stuck, taking forever to get close to the minimum. The goal is to find a learning rate that allows the model to move towards the minimum efficiently, without jumping around too much or taking too long.

3. **(15%)**
   - **What is the prior, likelihood, and posterior in Bayesian linear regression. [Explain the concept in detail rather than writing out the mathematical formula.]**

In Bayesian linear regression, we approach the problem of fitting a model to data by using probability to express uncertainty. Instead of just finding a single "best" set of parameters, we treat the parameters themselves as random variables. As we gather more data, we update our beliefs about what the parameters could be. The ideas of **prior**, **likelihood**, and **posterior** are key to understanding how the model works by combining what we already know with the new data we collect.

Prior

We think of the **prior** as our initial thoughts or beliefs about the model's parameters before we see any data. In linear regression, these parameters are usually the coefficients that link our input variables (like size, location, number of bedrooms in a house) to the output variable (like house price).

For example, if we're predicting house prices, we might start with the belief that larger houses are more expensive and that certain neighborhoods are pricier based on our past experience or knowledge. This initial belief about how each feature affects the price is captured in the prior distribution. Priors can be **informative**, where we have strong beliefs based on previous knowledge, or **non-informative**, where we don't have any strong initial beliefs and let the data speak for itself.

Likelihood

Once we have our prior beliefs, we collect some data. The **likelihood** is about how probable our observed data is, given specific values of the model parameters. In simpler terms, it measures how well our current model explains the data we've gathered.

Using the house price example, after collecting data on actual house prices and their features, the likelihood assesses how likely it is to see these prices if our model parameters (the coefficients) were a certain way. If our model predictions are close to the actual prices, the likelihood for those parameter values is high. If there's a big difference, the likelihood is low. This step is crucial because it tells us how well our model fits the data.

Posterior

The **posterior** is where everything comes together. It's our updated belief about the model parameters after considering both our prior beliefs and the new data. Essentially, the posterior combines the prior and the likelihood to give us a refined understanding of the parameters.

Using Bayes' Theorem, we update our prior beliefs with the evidence from the data (the likelihood) to form the posterior. So, if our prior thought was that house size has a positive effect on price and the data supports this, our posterior belief will strengthen that idea. Conversely, if the data shows something unexpected, like larger houses in a particular area being cheaper, our posterior will adjust to reflect this new information.

Putting It All Together

Start with the Prior: Begin with what we believe about the parameters based on previous knowledge or assumptions.

Collect Data and Calculate the Likelihood: Gather data and determine how likely this data is for different parameter values.

Update to the Posterior: Combine the prior and the likelihood to get the posterior, which is our updated belief after seeing the data.

- **What is the difference between Maximum Likelihood Estimation (MLE) and Maximum A Posteriori Estimation (MAP)? (Analyze the assumptions and the results.)**

Maximum Likelihood Estimation (MLE) and Maximum A Posteriori Estimation (MAP) are two fundamental methods used for estimating the parameters of statistical models. Both aim to find the most probable parameters given the observed data, but they approach this goal differently by incorporating different assumptions and yielding different results.

Core Difference

Maximum Likelihood Estimation (MLE) focuses entirely on the data that we have observed. The goal is to find the parameter values that make the observed data most likely. This is done by maximizing the **likelihood function**, which measures how probable the observed data is, given different parameter values. Essentially, MLE works by looking at the data alone, ignoring any prior knowledge we might have about the parameters.

On the other hand, **Maximum A Posteriori Estimation (MAP)** goes a step further. In addition to considering the likelihood from the observed data, MAP also takes into account any **prior knowledge** or beliefs we have about the parameters. It combines the likelihood with a **prior distribution**, which represents what we believe about the parameters before seeing the data. MAP then maximizes the **posterior distribution**, which is a combination of the data and the prior belief. This means MAP balances both the observed data and any prior information we might have.

Assumptions:

In **MLE**, the assumption is that the only thing that matters is the likelihood of the observed data. The parameters are treated as fixed but unknown, and we don't incorporate any prior knowledge about what those parameters could be. Essentially, MLE assumes that everything we know about the parameters comes from the data itself.

In **MAP**, we assume that we have some **prior belief** or information about the parameters before we see any data. This prior belief is expressed through a **prior distribution**, which reflects what we think about the parameters based on previous knowledge or assumptions. MAP's approach is to combine this prior belief with the data we observe (through the likelihood), meaning the final estimate should consider both the prior and the data. This makes MAP useful when we already have some knowledge about the parameters, even before looking at the new data.

Results:

With **MLE**, the result is the parameter values that make the observed data most likely. Since MLE focuses only on the likelihood, it's often described as being more **data-driven**. This means that if you have a large dataset, MLE works well because the data provides a lot of information about the parameters, and any prior beliefs would likely have minimal effect (if they were used at all).

MAP, however, gives you an estimate that maximizes the **posterior distribution**, which combines the likelihood from the data and the prior beliefs. The result tends to be more **conservative** if the prior is strong or if the dataset is small. MAP can be thought of as a **compromise** between what the data says and what the prior suggests. If the prior is highly informative and reliable, MAP will provide results that reflect both the prior knowledge and the observed data. On the other hand, if the prior is weak or flat (uninformative), the result will be closer to what MLE would give.