# CS2109s - Tutorial 4

Eric Han

Sep 20, 2023

# Annoucements

## Important admin

1. Midterm exam at Friday, 6 October (Week 7), 10:00 AM
2. PS4, try to use our functions where available - eg Q2.5, use `mean_squared_error`
3. Next tutorial, T5 is in Week 8.

## Tidbits from tutorials

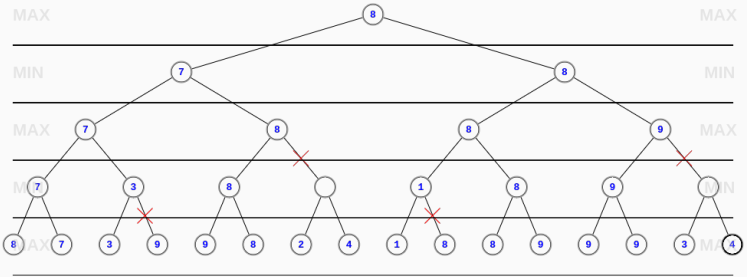


**Figure 1:** Alpha-Beta Answer (Credit MIT)

# Question 1

| | Income | Credit History | Debt | Decision |
|---|---|---|---|---|
| 0 | Over 10k | Bad | Low | Reject |
| 1 | Over 10k | Good | High | Approve |
| 2 | 0 - 10k | Good | Low | Approve |
| 3 | Over 10k | Good | Low | Approve |
| 4 | Over 10k | Good | Low | Approve |
| 5 | Over 10k | Good | Low | Approve |
| 6 | 0 - 10k | Good | Low | Approve |
| 7 | Over 10k | Bad | Low | Reject |
| 8 | Over 10k | Good | High | Approve |
| 9 | 0 - 10k | Bad | High | Reject |

**Question 1a-c [G]**

a. Construct the best decision tree to classify the final outcome (Decision).
b. First row is labelled wrongly `Reject > Approve`, construct the best decision tree.
c. What is the decision made by the decision tree in part (b) for a person with an **income over 10k**, a **bad credit history**, and **low debt**?

**Recap**

- What is a decision tree?
- How are the features selected (*best*)?
- How to calculate information gain?

**Question 1a-c [G]**

a. Construct the best decision tree to classify the final outcome (Decision).
b. First row is labelled wrongly `Reject > Approve`, construct the best decision tree.
c. What is the decision made by the decision tree in part (b) for a person with an **income over 10k**, a **bad credit history**, and **low debt**?

**Recap**

- What is a decision tree?
- How are the features selected (*best*)?
- How to calculate information gain?

$$IG(Y, X) = E(Y) - E(Y|X)$$

**Answer 1a**

E(Decision) = 0.8812908992306927

IG(Decision,X) = E(Decision) - E(Decision|X)

| | E(Decision\|X) | IG(Decision\|X) |
|---|---|---|
| Income | 0.879673 | 0.00161775 |
| Credit History* | 0 | 0.881291 |
| Debt | 0.879673 | 0.00161775 |

Credit History

|-- [Bad] Reject:3

+-- [Good] Approve:7

**Answer 1b**

**Choice 1 - Credit History**

E(Decision) = 0.7219280948873623

IG(Decision,X) = E(Decision) - E(Decision|X)

| | E(Decision\|X) | IG(Decision\|X) |
|---|---|---|
| Income | 0.68966 | 0.0322684 |
| Credit History* | 0.275489 | 0.446439 |
| Debt | 0.68966 | 0.0322684 |

## Choice 2 - Income

E(Decision) = 0.9182958340544894

IG(Decision,X) = E(Decision) - E(Decision|X)

| | E(Decision\|X) | IG(Decision\|X) |
|---|---|---|
| Income* | 0.666667 | 0.251629 |
| Debt* | 0.666667 | 0.251629 |

**Tree**

```
Credit History
|-- [Bad] Income
|   |-- [0 - 10k] Reject:1
|   +-- [Over 10k] Debt
|       +-- [Low] Approve:1 Reject:1
+-- [Good] Approve:7
```

**Answer 1c**

Cannot decide - the decision might either reject or accept the person.

**Question 1d-e [G]**

d. Following (c), what are some ways to to mitigate inconsistent data?

e. Derive a Tree with each leaf node representing a minimum of 3 training data points by pruning the tree you previously obtained in part (b). Which is pruned?

[1] Not always the best

**Question 1d-e [G]**

d. Following (c), what are some ways to to mitigate inconsistent data?

e. Derive a Tree with each leaf node representing a minimum of 3 training data points by pruning the tree you previously obtained in part (b). Which is pruned?

**Answer 1d**

- Regularize: Pruning to remove unnecessary branches and nodes, ie. Min-sample and Max-depth.
- Pre-processing: Remove outliers that create noise.
- Feature Selection: Select only relevant features, so less relevant features that could create inconsistencies are not part of the decision tree.
- Data Scarcity[1]: Collect more data on new features to clearly differentiate the inconsistent classes.

[1] Not always the best

**Answer 1e**

The outlier is probably the person with an income over 10k, a bad credit history, and low debt.

| | Income | Credit History | Debt | Decision |
|---|---|---|---|---|
| 0 | Over 10k | Bad | Low | Approve |
| 7 | Over 10k | Bad | Low | Reject |

# Question 2

$$\left(\begin{array}{ccc|c} 6 & 4 & 11 & 20 \\ 8 & 5 & 15 & 30 \\ 12 & 9 & 25 & 50 \\ 2 & 1 & 3 & 7 \end{array}\right)$$

a. Apply the Normal Equation formula to obtain a linear regression model that minimizes MSE of the data points.

b. Normal Equation needs the calculation of $(X^T X)^{-1}$. But sometimes this matrix is not invertible. When will that happen, and what should we do in that situation?

**Recap**

- What is the Normal Equation?
- What is the Gradient Descent?
- How does it compare with Gradient Descent?

**Answer 2a**

$w = (X^T X)^{-1} X^T Y$

$$X = \begin{pmatrix} 1 & 6 & 4 & 11 \\ 1 & 8 & 5 & 15 \\ 1 & 12 & 9 & 25 \\ 1 & 2 & 1 & 3 \end{pmatrix}, \quad Y = \begin{pmatrix} 20 \\ 30 \\ 50 \\ 7 \end{pmatrix} \rightarrow \quad X^T X = \begin{pmatrix} 4 & 28 & 19 & 54 \\ 28 & 248 & 174 & 492 \\ 19 & 174 & 123 & 347 \\ 54 & 492 & 347 & 980 \end{pmatrix},$$

$$(X^T X)^{-1} = \begin{pmatrix} 5.5 & -5.75 & -4 & 4 \\ -5.75 & 7.5 & 6.5 & -5.75 \\ -4 & 6.5 & 14 & -8 \\ 4 & -5.75 & -8 & 5.5 \end{pmatrix}, \quad X^T Y = \begin{pmatrix} 107 \\ 974 \\ 687 \\ 1941 \end{pmatrix}$$

$$\rightarrow (X^T X)^{-1} X^T Y = \begin{pmatrix} 4 \\ -5.5 \\ -7 \\ 7 \end{pmatrix} \rightarrow \hat{y} = 4 - 5.5x_1 - 7x_2 + 7x_3$$

**Answer 2b**

- Matrix $X^T X$ is not invertible
  - Some rows or columns in the dataset matrix are linearly dependent
    - Columns are linearly dependent $>$ Highly correlated or redundant features
    - Rows are linearly dependent $>$ collinear or duplicates
- Matrix $X^T X$ is ill-conditioned (almost singular)
  - Some rows or columns in the dataset matrix are *almost* linearly dependent
  - Not enough data points to cover the number of features

In such instances, gradient descent can be used to arrive at weights for the linear regression model that minimise the cost function.

**Intuition**: If invertible, there is a single solution, otherwise there are multiple solutions.

# Question 3 [G]

For Linear Regression, there are two popular cost functions, **Mean Squared Error**: $L(y, \hat{y}) = \frac{1}{2}(y - \hat{y})^2$ and **Mean Absolute Error**: $L(y, \hat{y}) = \frac{1}{2}|y - \hat{y}|$

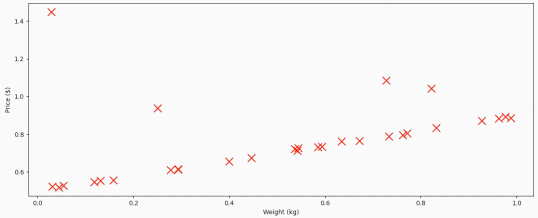


**Figure 2:** Scatter plot of the actual weight of meat at NTUC (x) and its price (y)

a. Justify your choice of cost function for this problem.
b. Provide examples of cost functions that are better suited to handle outliers?

## Recap

- What is the difference between / when should we use MSE and MAE?
- How does it relate to L2, L1 loss?

## Recap

- What is the difference between / when should we use MSE and MAE?
- How does it relate to L2, L1 loss?

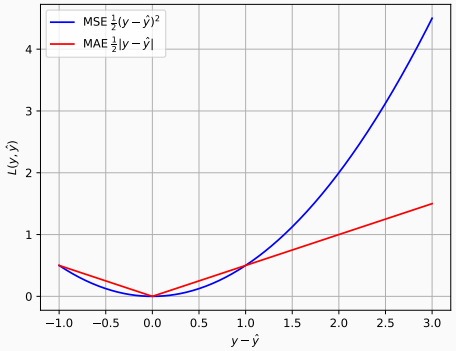


**Figure 3:** Cost Functions Plot

L1/L2 are technically vector lengths but are commonly interchanged with MAE/MSE.

## Answer 3a

Two examples are shown below when the cost functions are MAE and MSE respectively.

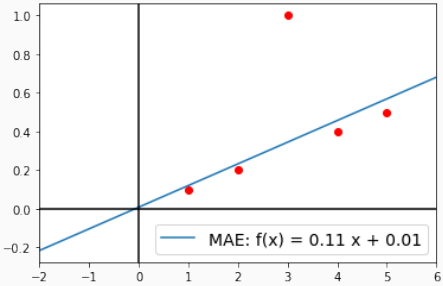


**Figure 4:** MAE

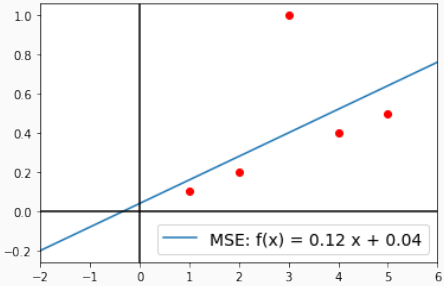


**Figure 5:** MSE

Note: As shown in these examples, outliers can have a greater impact on MSE than MAE, even if the y-values are between 0 and 1.

**Answer 3b**

**Huber loss** is a combination of MSE and MAE and is defined as:

$$L(y, \hat{y}) = \begin{cases} \frac{1}{2}(y - \hat{y})^2 & \text{for } |y - \hat{y}| \leq \delta \\ \delta \cdot |y - \hat{y}| - \frac{1}{2}\delta^2 & \text{otherwise} \end{cases}$$

where $\delta$ is a transition threshold between the MSE and MAE behaviors.

**Log-cosh loss** is defined as:

$$L(y, \hat{y}) = \log(\cosh(y_i - \hat{y}_i))$$

For small values of x, $\log(\cosh(x)) \approx \frac{1}{2}x^2$, which is similar to MSE. For larger values of x, $\log(\cosh(x)) \approx |x| - \log(2)$, which is similar to MAE. Log cosh approximates MSE and MAE and is similar to the Huber loss function.

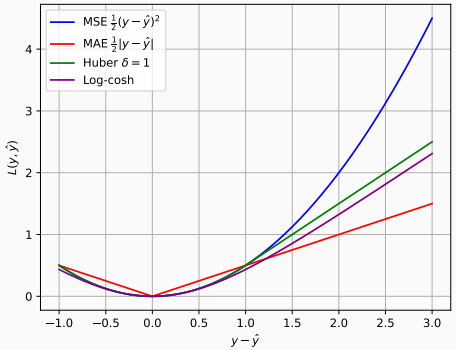


**Figure 6:** Cost Functions Plot

Huber loss and Log cosh loss are more robust to outliers compared to MSE and MAE because they don't give as much weight to extreme values. This makes them useful in cases where the presence of outliers might negatively impact model performance if using MSE or MAE.

# Question 4 [G]

Given a simple function $y = x^2$, we know the gradient is $\frac{dy}{dx} = 2x$. As such, the minimum of this function is 0.

a. Compute over 5 iterations for $\alpha \in \{10, 1, 0.1, 0.01\}$ in tabular format.
b. During the course of training for a large number of epochs/iterations, what can be done to the value of the learning rate *alpha* to enable better convergence?

**Recap**

- What is gradient descent?
- What is the equation to perform the updates?
- Why do we need a learning rate $\alpha$, and what is its use?

**Answer 4a**

| t | 10.0 | 1.0 | 0.1 | 0.01 |
|---|---|---|---|---|
| 0 | 5 | 5 | 5 | 5 |
| 1 | -95 | -5 | 4 | 4.9 |
| 2 | 1805 | 5 | 3.2 | 4.802 |
| 3 | -34295 | -5 | 2.56 | 4.70596 |
| 4 | 651605 | 5 | 2.048 | 4.61184 |
| 5 | -12380495 | -5 | 1.6384 | 4.5196 |

**Answer 4b**

Learning rate scheduler, $\alpha$ can be decreased through the course of training - larger values at the start, smaller values near the end:

- **Exponential Decay:** $\alpha(t) = \alpha_0 e^{-kt}$

# Bonus Qn

To help you further your understanding, not compulsory; Work for Snack/EXP!

**Tasks**

1. Implement `create_tree(tree, df, parent=None, action = '')` to solve Q1 to get the following trace; `IG` implemented using pandas.
   1.1 Some code implemented in https://github.com/eric-vader/CS2109s-2324s1-bonus

```
Credit History
|-- [Bad] Income
|   |-- [0 - 10k] Reject:1
|   +-- [Over 10k] Debt
|       +-- [Low] Approve:1 Reject:1
+-- [Good] Approve:7
```

# Buddy Attendance Taking

Take Attendance for your buddy: https://forms.gle/Ckkq639TNwWEx3NT6

1. Random checks will be conducted - `python ../checks.py TG0`

**Figure 7:** Buddy Attendance