Introduction to Deep Learning

Review of Logistic Regression

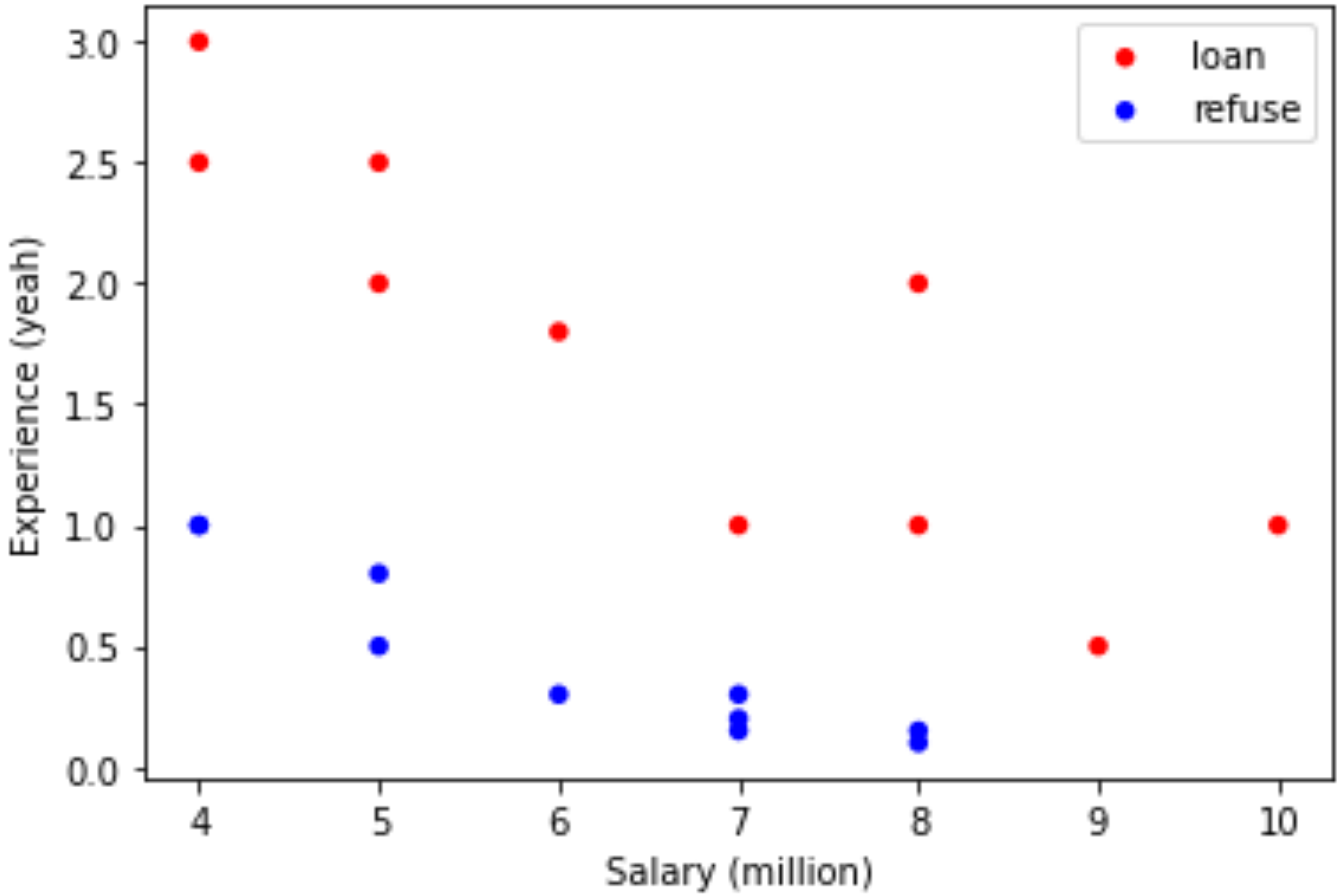
Example

- Problem: decision supporting for loan program
- Input: Having data about salary and working time of employees as in table 1 below

Salary	Working Time	Loan Decision
10	1	1
9	0.5	1
5	2	1
…	…	…
8	0.1	0
6	0.3	0
7	0.15	0
…	…	…

Table 1: Dataset for loan decision

Visualization of table 1 data



Model Definition

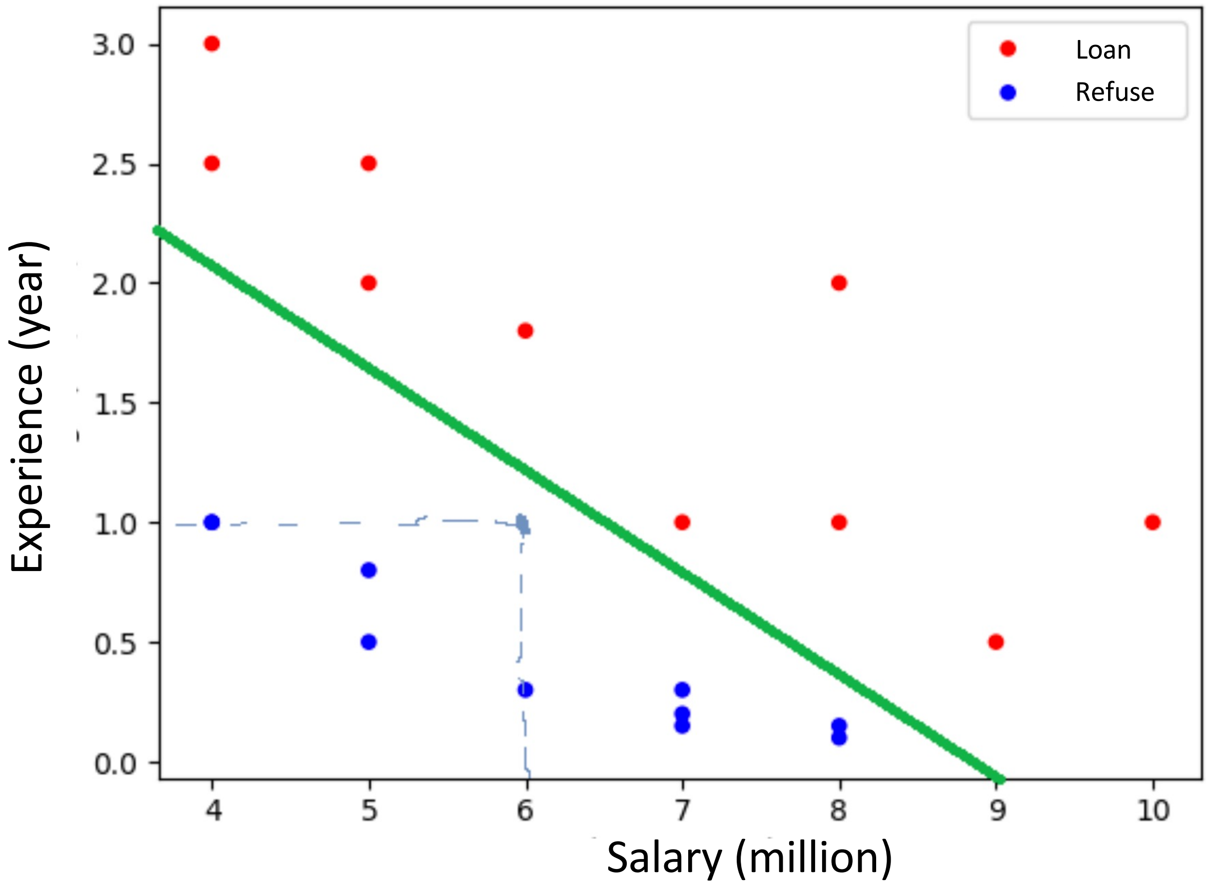
- With the row i in the data table, let $x_1^{(i)}$ be the salary and $x_2^{(i)}$ be the working time of the profile i

- Prediction model is defined as follows:

$$\hat{y}_i = w_0 + w_1 * x_1^{(i)} + w_2 * x_2^{(i)}$$

Model Visualization

Separation line (green) and new data point prediction (at 6 salary)



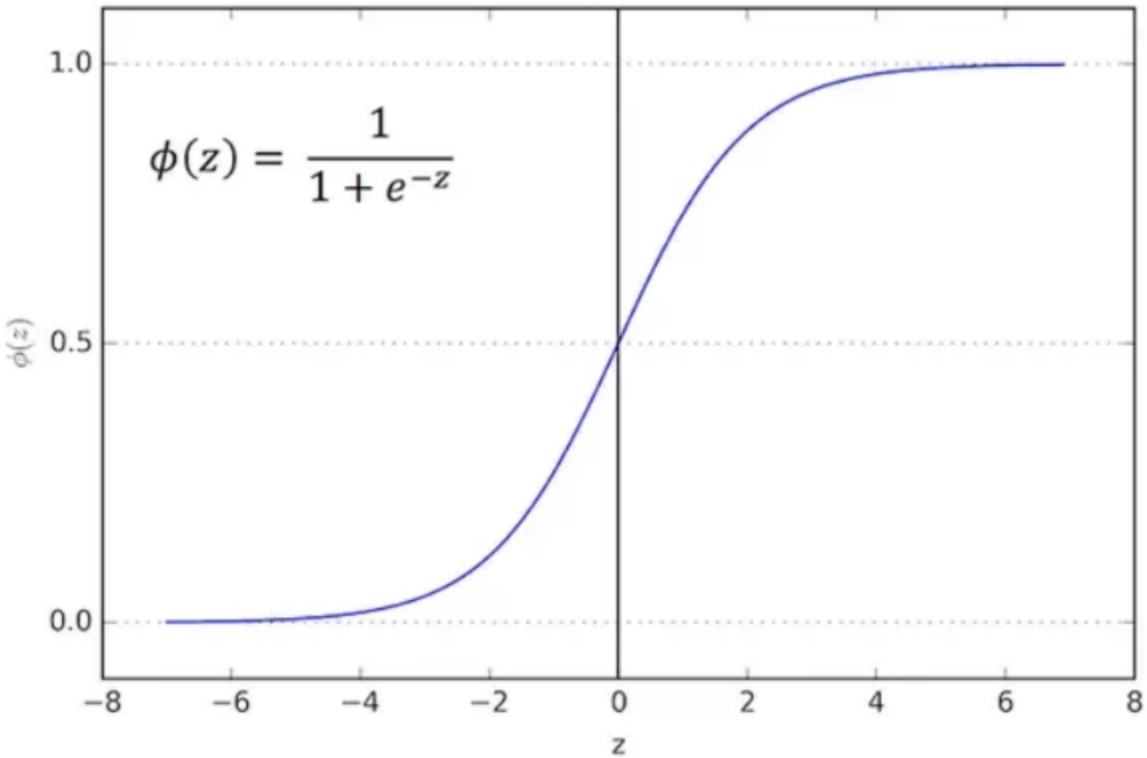
→ Result: profile at salary = 6 is not loaned

More Constraint

- Requirement: estimate the probability that a new profile should be loaned or not
- Output:
  - if the estimated loan probability >= threshold t, then the new profile should be loaned
  - Otherwise, it should be refused

Sigmoid function

- Continuous function with real values in the interval (0,1)
- Have derivative at every point (for applying gradient descent)



Model Definition (next)

- Estimated loan probability $\hat{y}_i$ is defined as follows:

$$\hat{y}_i = \sigma(\hat{y}_i) = \sigma(w_0 + w_1 * x_1^{(i)} + w_2 * x_2^{(i)})$$

Model Definition (next)

- In detail, the estimated loan probability $\hat{y}_i$ is written as:

$$\hat{y}_i = \sigma(w_0 + w_1 * x_1^{(i)} + w_2 * x_2^{(i)}) = \frac{1}{1 + e^{-(w_0 + w_1 * x_1^{(i)} + w_2 * x_2^{(i)})}}$$

Loss Function

- Consider the probability that the model predicts that the profile i will be loaned as follows:

$$\mathrm{p}(x^{(i)} = 1) = \hat{y}_i$$

- Consider that the probability that the model predicts that the profile i will not be loaned as follows:

$$\mathrm{p}(x^{(i)} = 0) = 1 - \hat{y}_i$$

- In total, we have:

$$\mathrm{p}(x^{(i)} = 1) + \mathrm{p}(x^{(i)} = 0) = 1$$

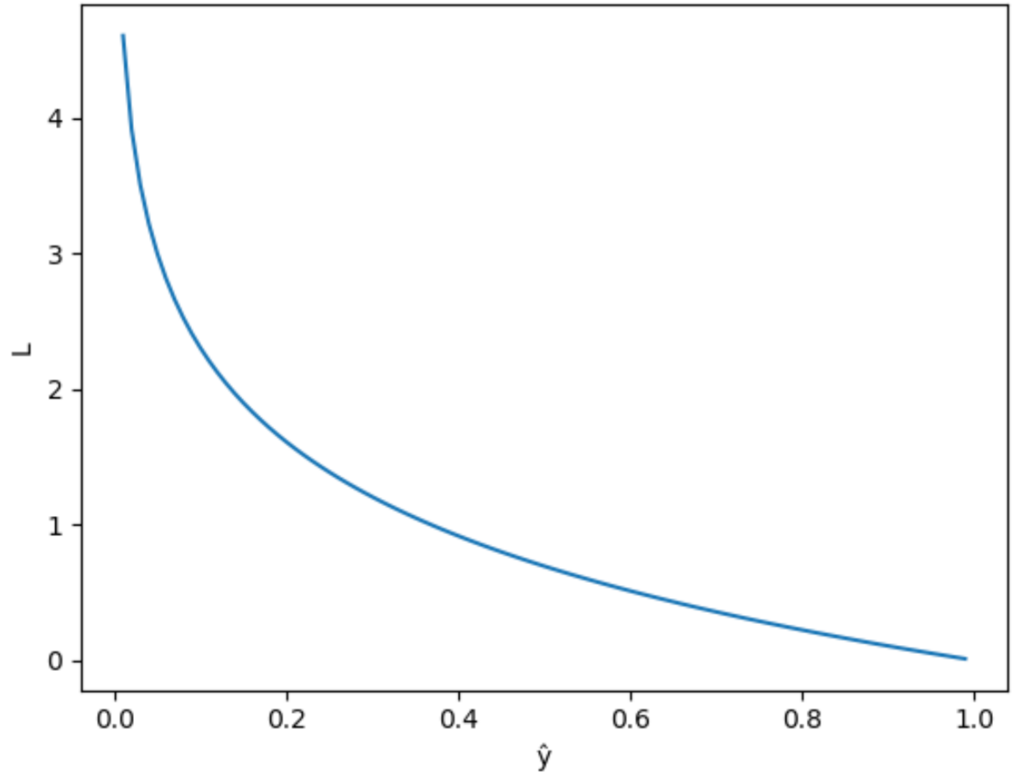
Loss Function

- For each data point $(x^{(i)}, y_i)$, loss function L is defined as:

$$L = -(y_i * log(\hat{y}_i) + (1 - y_i) * log(1 - \hat{y}_i))$$

Loss Function

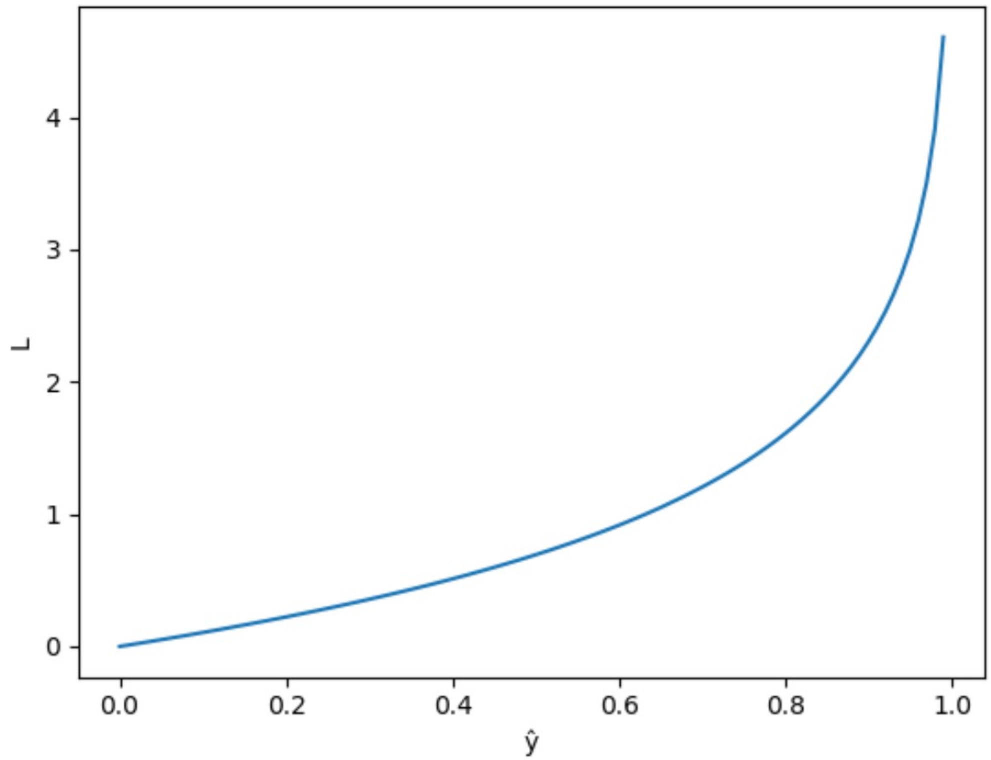
$$L = -(y_i * log(\hat{y}_i) + (1 - y_i) * log(1 - \hat{y}_i))$$



loss function when $y_i = 1$

Loss Function

$$L = -(y_i * log(\hat{y}_i) + (1 - y_i) * log(1 - \hat{y}_i))$$



loss function when $y_i = 0$

Loss Function

- For all data points, loss function J is defined as:

$$J = -\frac{1}{N} * \sum_{i=1}^{N} (y_i * log(\hat{y}_i) + (1 - y_i) * log(1 - \hat{y}_i))$$

→ J is called binary cross-entropy loss

Loss Function

- Given the loss function J:

$$J = -\frac{1}{N} * \sum_{i=1}^{N} (y_i * log(\hat{y}_i) + (1 - y_i) * log(1 - \hat{y}_i))$$

→ Apply gradient descent algorithm to find parameters $\{w_0, w_1, w_2\}$ which minimize J

New profile prediction

- Given a new profile (x_{new}, y_{new}) → calculate predicted loan probability $\hat{y}_{new}$ using found parameters {w_0, w_1, w_2}

- Loan decision is defined as follows:

If $\hat{y}_{new}$ >= threshold t, the profile is loaned, otherwise, it is refused

Tool Installation

- Online with Google colab
- Local with Anaconda
- IDE: Jupyter notebook, VS code, spyder, etc.
- Framework: Pytorch, Keras, Tensorflow, etc.

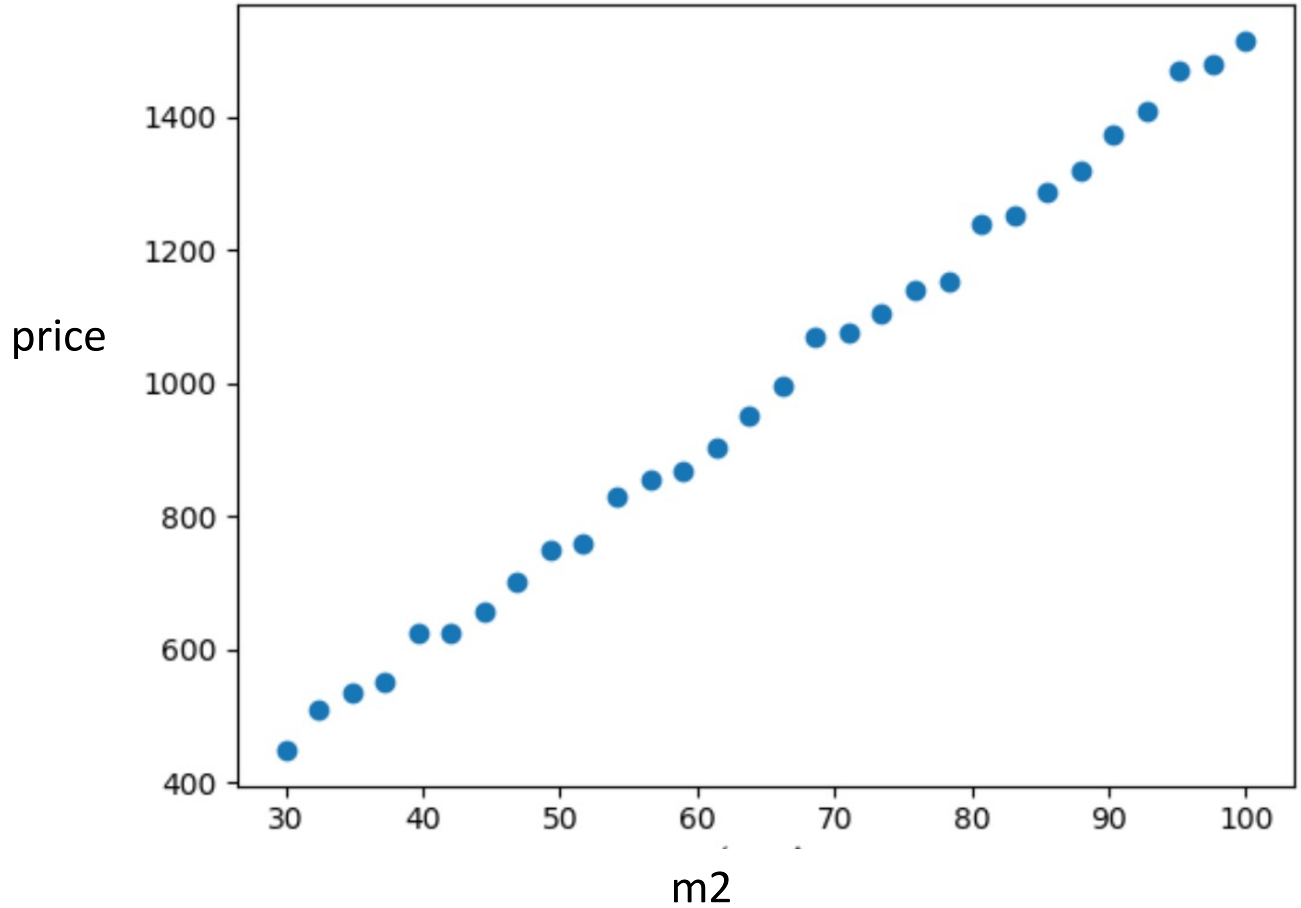
→ Do it by yourself to complete the labworks

Exercises 1

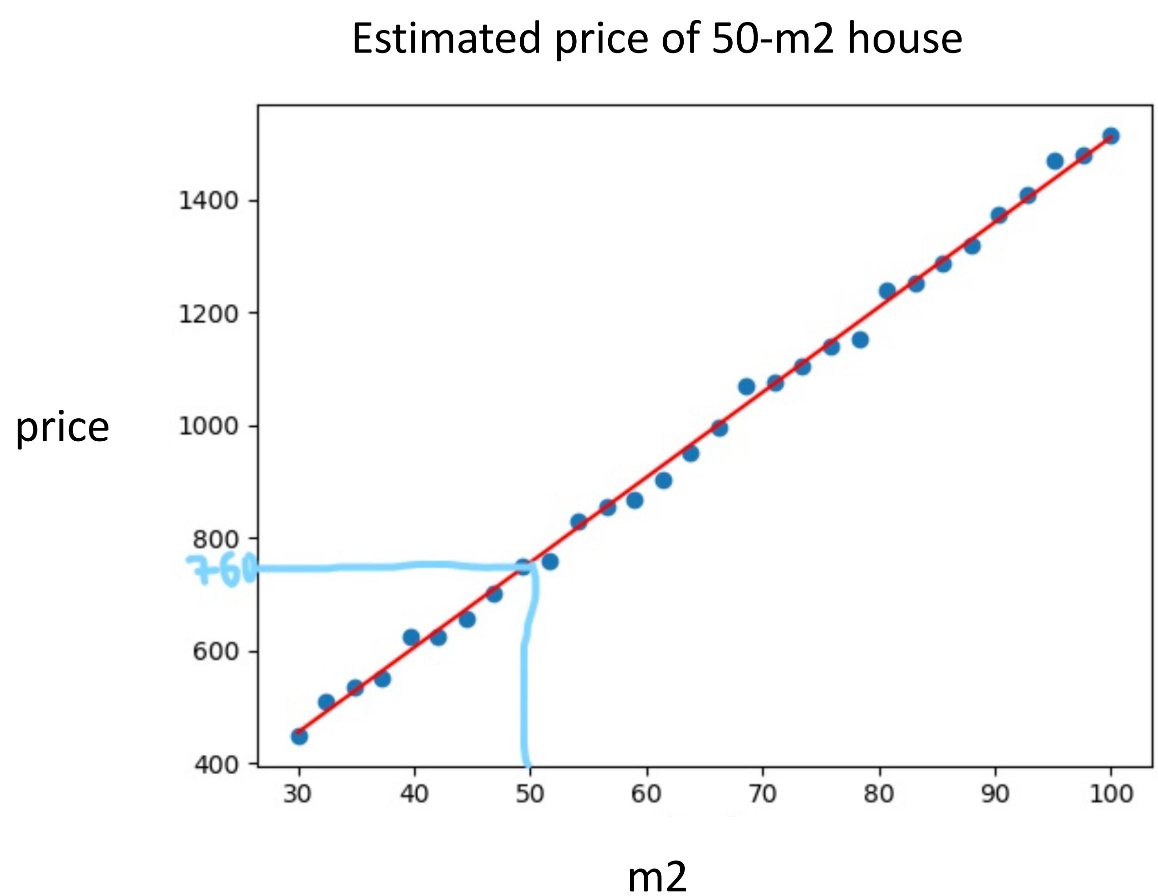
- Given data of exercise_1 in the google drive folder of the course
- Draw the graph of exercise 1 in the slide 19, then add the red line similar to the slide 20
- Upload your codes in the google drive folder of the course

Graph of exercise 1

Relationship between house selling price
and house area



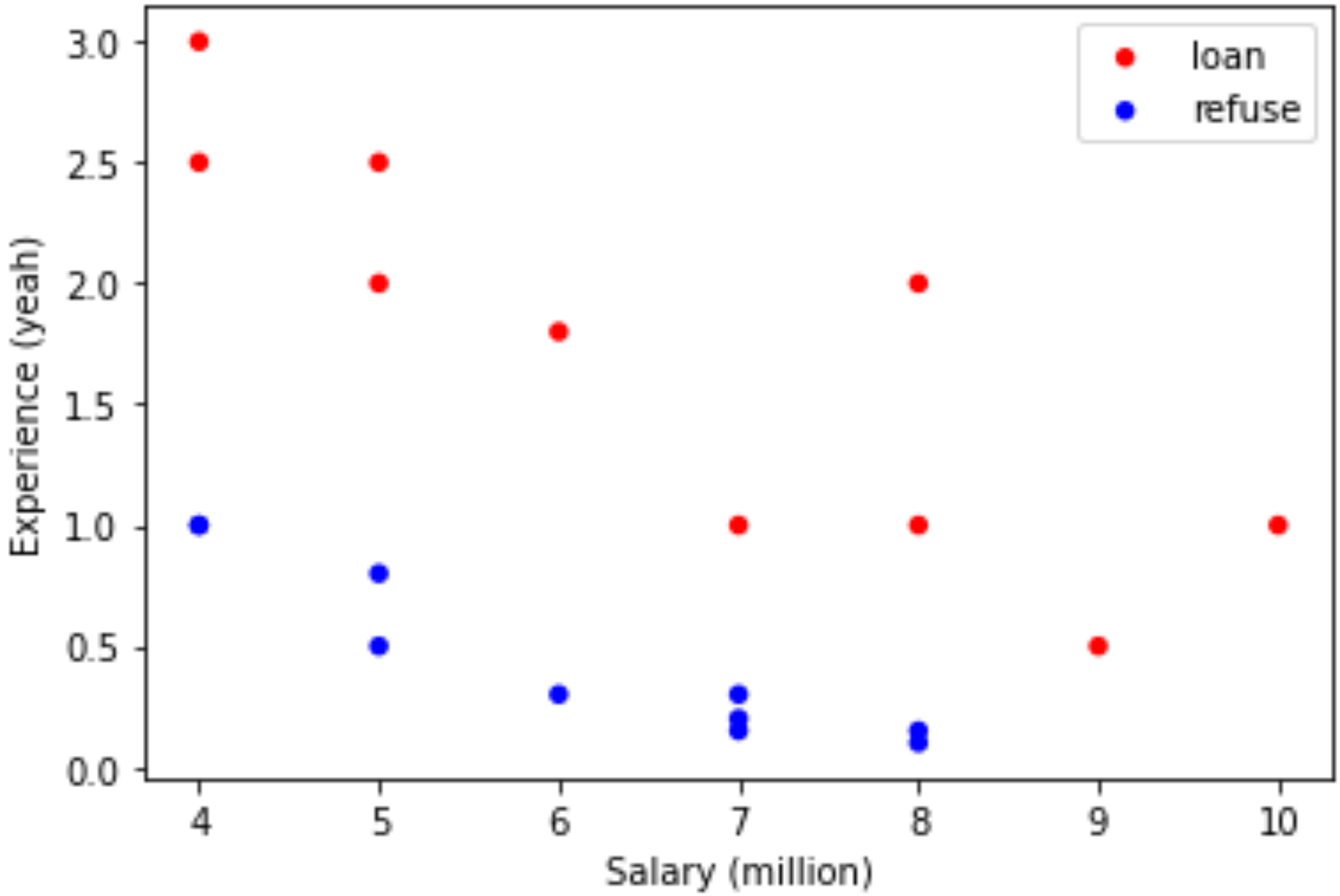
Graph of exercise 1



Exercises 2

- Given data of exercise_2 in the google drive folder of the course
- Draw the graph of exercise 2 in slides 22, then add the green line similar to the slide 23
- Upload your codes in the google drive folder of the course

Graph of exercise 2



Graph of exercise 2

Separation line (green) and new data point prediction (at 6 salary)