

Math Lab #2: Final Exam Score Prediction



Sunglok Choi, Assistant Professor, Ph.D.
Computer Science and Engineering Department, SEOULTECH
sunglok@seoultech.ac.kr | <https://mint-lab.github.io/>

Overview

- **Prerequisite**

- Anaconda (Individual Edition)

- **Practice) Final Exam Score Prediction**

- The given data
 - Expected results
 - Practice with the skeleton code
 - Step #1) Find a line

- **Assignment**

- Mission: Complete the given skeleton code

Practice) Final Exam Score Prediction

- The given data (file: data/class_score_en.csv)

midterm (max 125), final (max 100)

113, 86

104, 83

110, 78

101, 79

101, 77

103, 76

71, 94

102, 71

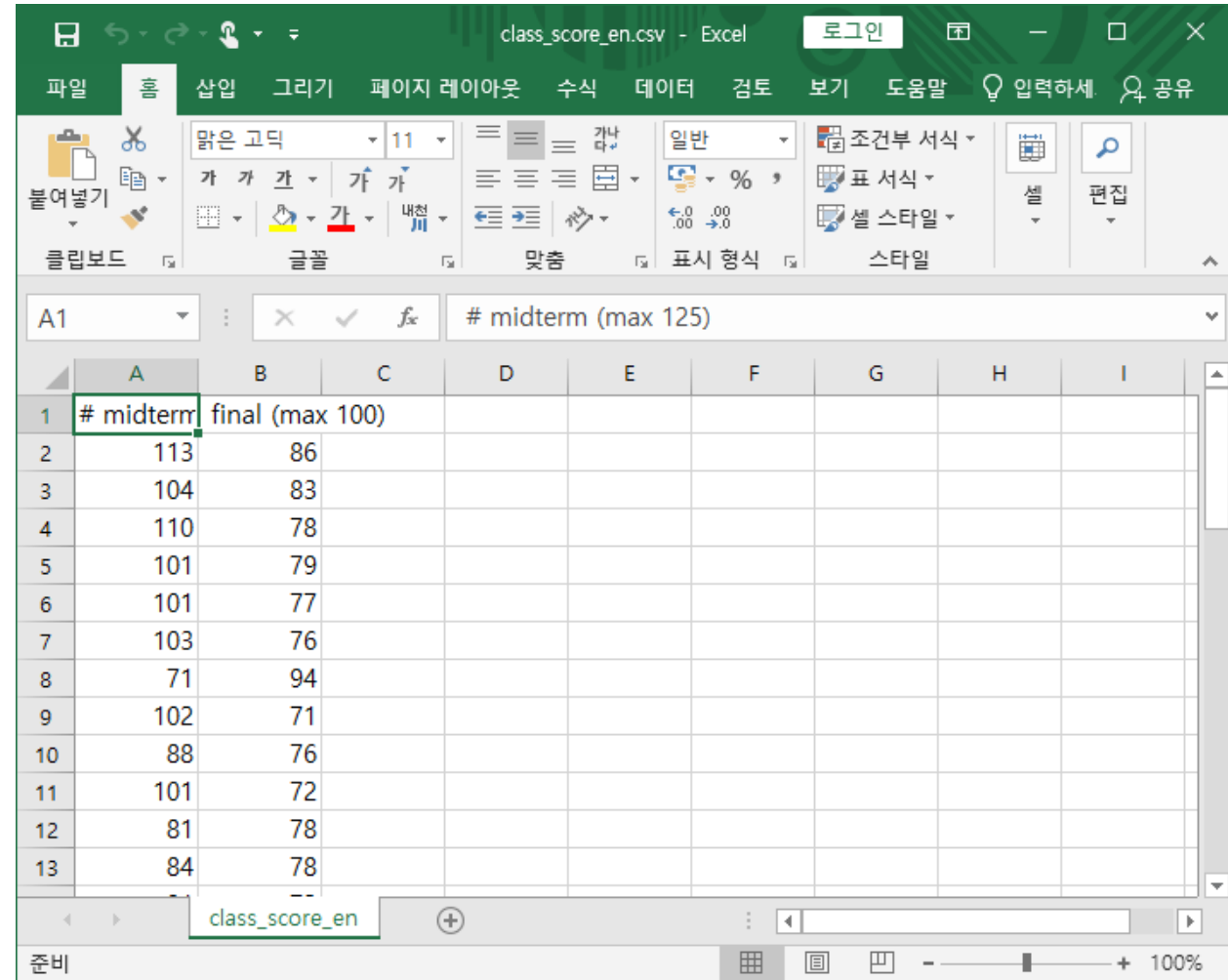
88, 76

101, 72

81, 78

84, 78

...



	# midterm (max 125)	final (max 100)
1	113	86
2	104	83
3	110	78
4	101	79
5	101	77
6	103	76
7	71	94
8	102	71
9	88	76
10	101	72
11	81	78
12	84	78
13		

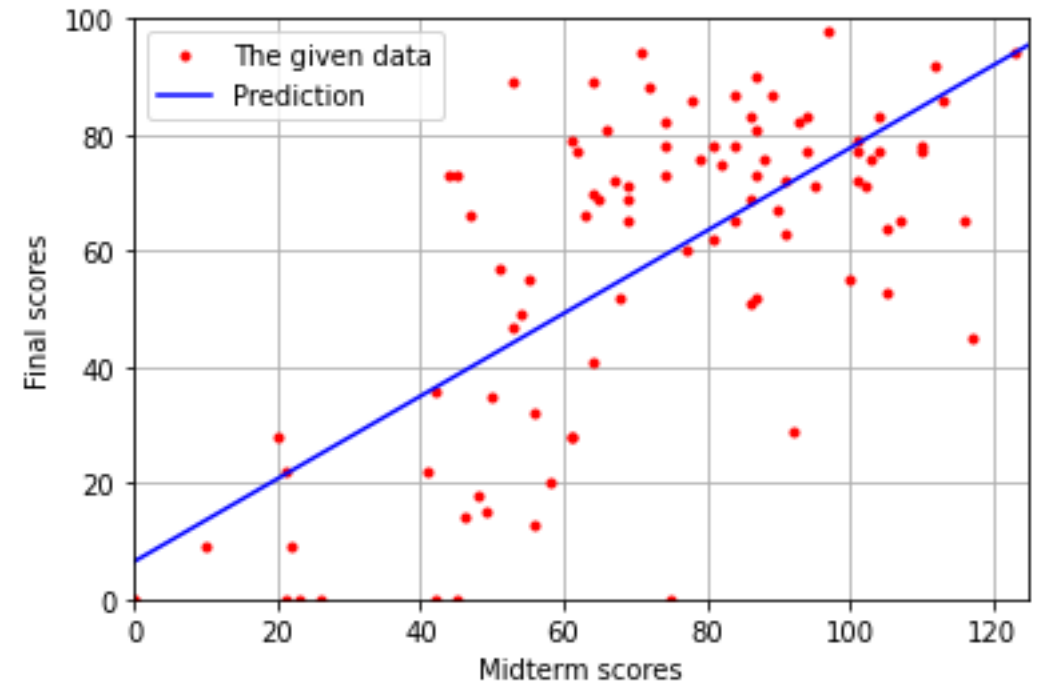
Practice) Final Exam Score Prediction

- Expected results
 - Problem
 - Given: The midterm exam score
 - Output: The final exam score
 - Solution: Line fitting
 - Examples
 - Q) Please input your midterm score? 10
 - A) Your final score is expected to 13.608.

 - Q) Please input your midterm score? 40
 - A) Your final score is expected to 34.970.

 - Q) Please input your midterm score? 90
 - A) Your final score is expected to 70.573.

 - Q) Please input your midterm score? 120
 - A) Your final score is expected to 91.934.



Practice) Final Exam Score Prediction

- The given skeleton code

(class_score_predict_skeleton.py)

- Step #1) Find a line

$$\begin{bmatrix} x_1 & 1 \\ x_2 & 1 \\ \vdots & \vdots \\ x_n & 1 \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$

(Ax=b)

```
import numpy as np
import matplotlib.pyplot as plt

if __name__ == '__main__':
    midterm_range = np.array([0, 125])
    final_range = np.array([0, 100])

    # Load score data
    class_kr = np.loadtxt('data/class_score_kr.csv', delimiter=',')
    class_en = np.loadtxt('data/class_score_en.csv', delimiter=',')
    data = np.vstack((class_kr, class_en))

    # Estimate a line, final = slope * midterm + y_intercept
    line = [0, 0] # TODO: Please find the best [slope, y_intercept] from 'data'

    # Predict scores
    final = lambda midterm: line[0] * midterm + line[1]
    while True:
        given = input('Q) Please input your midterm score (Enter or -1: exit)? ')
        if given == '' or float(given) < 0:
            break
        print(f'A) Your final score is expected to {final(float(given)):.3f}.')

    # Plot scores and the estimated line
    plt.figure()
    plt.plot(data[:,0], data[:,1], 'r.', label='The given data')
    plt.plot(midterm_range, final(midterm_range), 'b-', label='Prediction')
    ...
```

Assignment

- Mission
 - Complete the given skeleton code (`class_score_predict_skeleton.py`)
 - Submit your code (`class_score_predict.py`) and its prediction plot (`class_score_predict.png`)
- Condition
 - Please follow the above filename convention.
 - Please do not use `numpy.polyfit()`, `numpy.linalg.lstsq()`, and `sklearn.linear_model.LinearRegression`.
 - You already have power to implement it using `numpy.linalg.pinv()`.
 - You **can** start from scratch (without using the given skeleton code).
 - However, you **should** use the given data.
 - You **can** freely change the given skeleton code if necessary.
- Submission
 - Deadline: **October 16, 2024 23:59** (**firm deadline**; no extension)
 - Where: e-Class > Assignments
 - Score: Max 10 points