Lab#2: Linear regression in TensorFlow

Prepared by: Teeradaj Racharak (r.teeradaj@gmail.com)

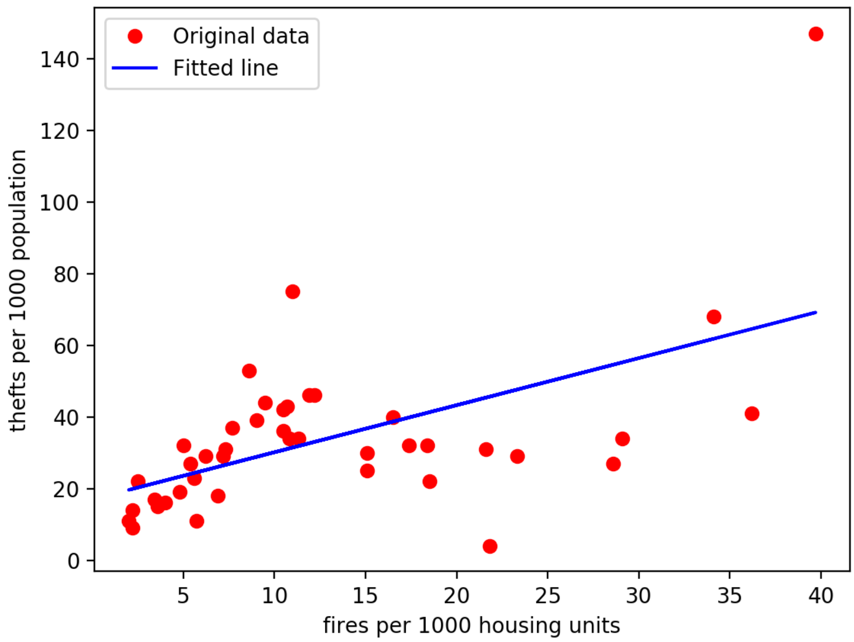
Scenario: We often hear that insurance companies using factors such as the number of fire and theft in a neighborhood to calculate how dangerous the neighborhood is. My question is: “is there a relationship between the number of fire and theft in a neighborhood? If so, can we find it?” In other words, can we find a function h such that if X is the number of fire and Y is the number of theft, then: $Y = h(x)$?

Suppose we have the **dataset collected by U.S. Commission on Civil Rights** as provided with this lab. The dataset description is as follows:
X = fire per 1000 housing units
Y = theft per 1000 population
within the same zip code area, where total number of zip code areas is 42.

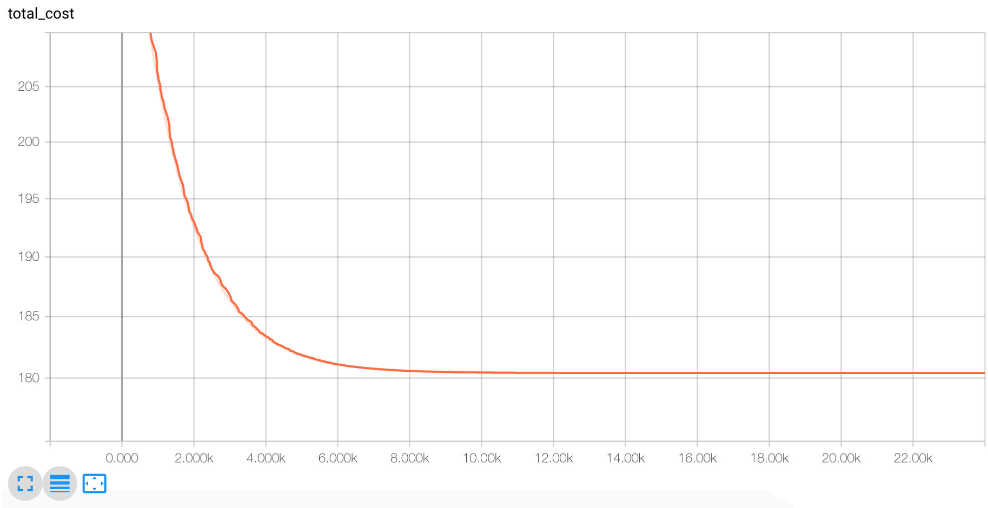
Problem 1: Assume that the relationship between the number of fire and theft are linear *i.e.* $h_\theta(x) = \theta_0 + \theta_1 x$. Let’s write a program that uses:

- batch gradient descent for training the parameters θ_0, θ_1
- mean squared error as the cost function

Answer the values of θ_0, θ_1 and plot a graph as following after training for 30,000 epochs.



Problem 2: Now, we are interested to see how fast the gradient descent is converged. We can try to log this information as an event file and use TensorBoard to visualize it later. As a hint, you need to use **tf.summary.scalar** for logging such information over each iteration. The following figure displays what we can see on TensorBoard.



Problem 3: Based on the same problem, try to implement normal equations and compare its estimated θs with ones computed from the above.

Problem 4: Reimplement the program in Problem 1 by basing on stochastic gradient descent and log the mean squared error for each training epoch as a scalar summary on TensorBoard. Compare its training epoch with the one in batch gradient descent.

Problem 5: Reimplement the program in Problem 1 by basing on mini-batch gradient descent (use 50 as its batch size) and log the mean squared error for each training epoch as a scalar summary on TensorBoard. Compare its training epoch with the ones in batch gradient descent and stochastic gradient descent implemented in Problem 4.

Problem 6 (Bonus): Try to reimplement the program in Problem 1 from scratch (*i.e.* without using TensorFlow). You may see its algorithm in the slide to get some ideas.