TensorFlow is the most common library used in deep learning. It can be used for other machine learning tasks as well.
Today we will do simple exercises in order to understand the basics of the library. We will begin along the following tutorial:
https://www.tensorflow.org/tutorials/customization/basics

This exercise is divided into two parts, one for TF1.x, and the other for TF2.0.
Because most companies have not yet migrated to TF2.0, it is important you will be familiar with both.
The main difference between TF1.0 and TF2.0 is that in TF1.0 you had to first define a computational graph, and then run a session where data is fed into your graph and outputs are calculated.
In TF2.0 eager_execution() is enabled by default, which means variables are calculated and assigned a value when they are created and running a session happens in the background.
In case you have Tensorflow 2.0, you can `import tensorflow.compat.v1 as tf` and immediately run the method `tf.disable_v2_behavior()` in order to use the old version of TensorFlow.
Repeat the introduction of this exercise for both TF1.0 and TF2.0, create the Linear Regression Model at the end with the TF version of your choice.

**Introduction**:

1. Make sure tensorflow is installed on your enviorement: 'pip install tensorflow'

   (note: if you have a compatible gpu, 'pip install tensorflow-gpu' instead)

2. import tensorflow as tf

3. Check your version of tf (`print(tf.__version__)`)

4. Create two constant nodes (node1, node2) with the values [1,2,3,4,5] and [1,1,2,3,5]

5. Perform an element-wise multiplication of the two nodes and store it to node3
   (TF1: Create a TensorFlow Session and run it to produce the answer
   TF2: Print the value of node3 (use the .numpy() method)).

6. Sum the values of the elements in node3, store result in node4.

TensorFlow 1.x only:

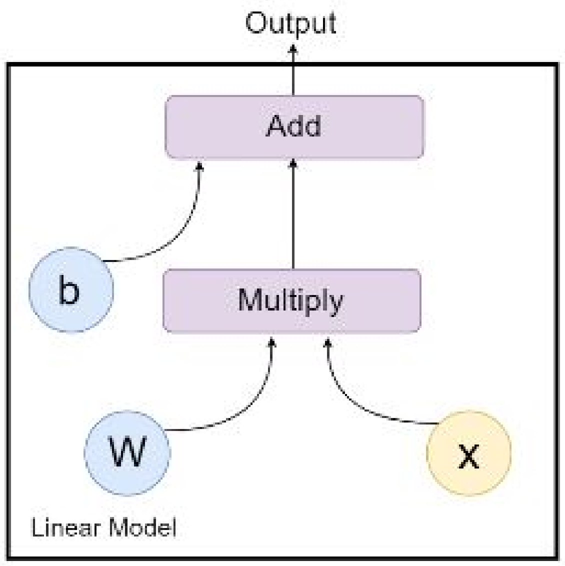
7. Create two placeholders (X,Y), add them using a feed_dict in the session run function
8. Create two variables (W,b) and initialize their values to 7,8, add them and run a session, don't forget to initialize all variables before running the session

**Linear Regression**
In this exercise you will define a simple Linear Regression model with TensorFlow low-level API.

for full instructions and notes by google:
https://www.tensorflow.org/tutorials/customization/custom_training

9. Load the data from the file "data_for_linear_regression_tf.csv".

As we plan models in TensorFlow it is recommended to have the graph representation of the model in mind. for a Linear Model y = Wx+b the graph looks like this:



Each shape in the graph represents a node in the model.
We will define either an attribute or a method in our model for each node.

10. Define a class called MyModel():
11. The class should have two Variables (W and b), and a call method* that returns the model's output for a given input value. * - "def __call__(self, x):"

Hint:

[redacted]

Hint:

[redacted]

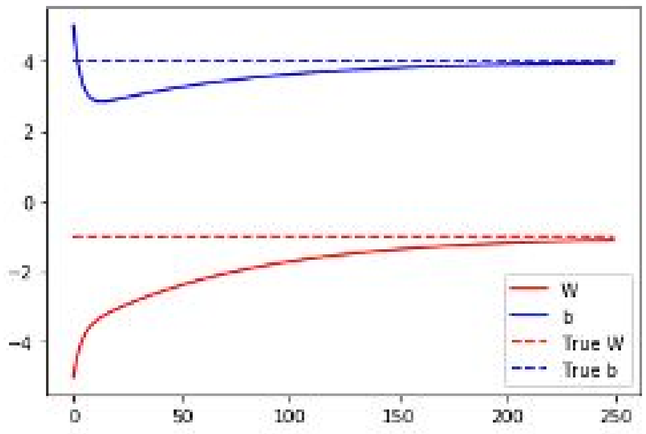
12. Define a loss method that receives the predicted_y and the target_y as arguments and returns the loss value for the current prediction.

13. Define a train() method that does a single train cycle of your model (e.g 1 epoch)
    A. use tf.GradientTape() and the loss function that you have defined to record the loss as the linear operation is processed by the network.
    B. use the tape.gradient() method to retrieve the derivative of the loss with respect to W and b (dW, db) and update W and b accordingly.

Hint:

[redacted]

14. Now, use the data to train and test your model:
    A. Train your model for 1000 epochs, with learning_rate 0.1
    B. Save your model's W and b after each epoch, store results in a list for plotting.
    C. Print the current W, b and loss values after training


W
b
True W
True b