Unit 6 Softmax Regression Part 03 Introduction to TensorFlow

TFIP-AI Artificial Neural Networks and Deep Learning

Motivating problem

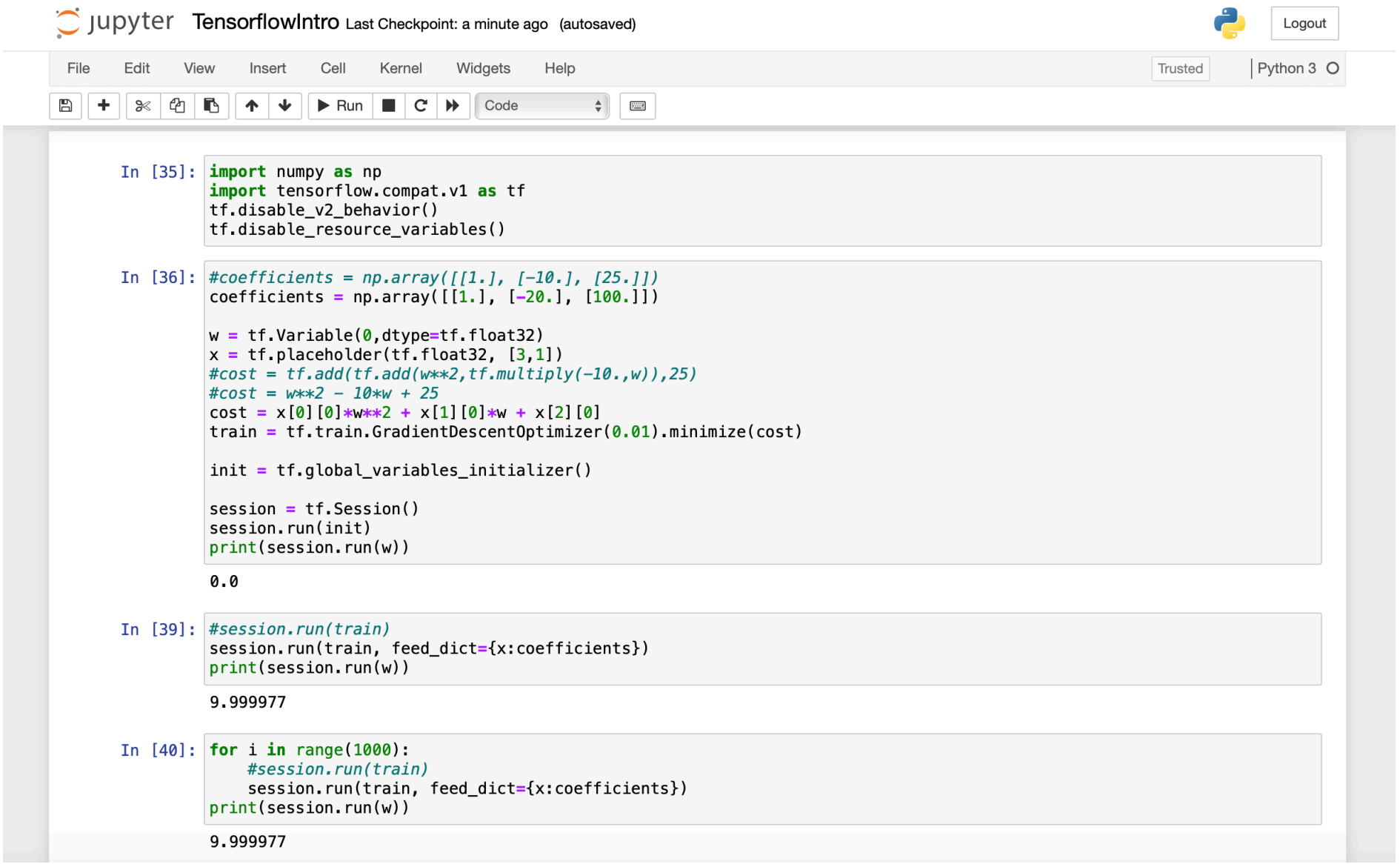
- Cost:

$$\begin{aligned}\mathbf{J}(w) &= w^2 - 10w + 25 \\ &= (w-5)^2\end{aligned}$$

$$w = 5$$

$$\mathbf{J}(w, b)$$

An Example with TensorFlow version 1



An Example with TensorFlow version 2

jupyter TensorflowIntro_v02 Last Checkpoint: Last Saturday at 22:56 (unsaved changes) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

Run Code

In [1]:
```
import numpy as np
import tensorflow as tf
```

In [2]:
```
w = tf.Variable(tf.zeros([1], dtype=tf.float32))
x = tf.Variable([1., -10., 25.], dtype=tf.float32)
#x = tf.Variable([1., -20., 100.], dtype=tf.float32)

with tf.GradientTape() as tape:
    #cost = tf.add(tf.add(w**2,tf.multiply(-10.,w)),25)
    #cost = w**2 - 10*w + 25
    cost = x[0]*w**2 + x[1]*w + x[2]

grad = tape.gradient(cost, [w])
print(grad)

optimizer = tf.optimizers.SGD(learning_rate=0.001)
optimizer.apply_gradients(zip(grad, [w]))

print(w)
print(cost)
```

```
[<tf.Tensor: id=82, shape=(1,), dtype=float32, numpy=array([-10.], dtype=float32)>]
<tf.Variable 'Variable:0' shape=(1,) dtype=float32, numpy=array([0.01], dtype=float32)>
tf.Tensor([25.], shape=(1,), dtype=float32)
```

In [3]:
```
for i in range(1000):
    optimizer.apply_gradients(zip(grad, [w]))
    #cost = tf.add(tf.add(w**2,tf.multiply(-10.,w)),25)
    #cost = w**2 - 10*w + 25
    cost = x[0]*w**2 + x[1]*w + x[2]
    if cost < 0.01:
        break

print(w)
print(cost)
```

```
<tf.Variable 'Variable:0' shape=(1,) dtype=float32, numpy=array([4.9000173], dtype=float32)>
tf.Tensor([0.00999832], shape=(1,), dtype=float32)
```

Another Example with TensorFlow version 2

jupyter TensorflowIntro_v02_01 Last Checkpoint: Last Sunday at 10:28 (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

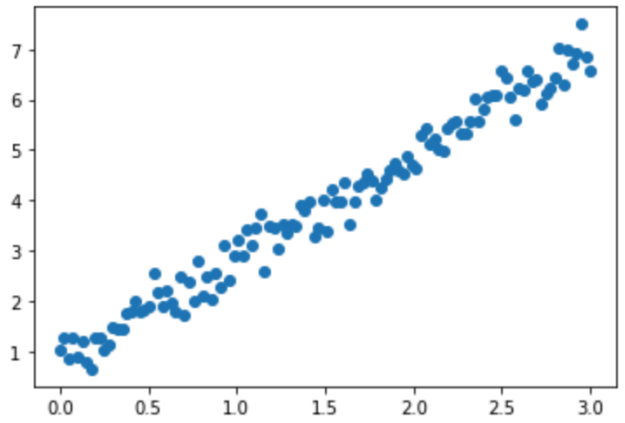
Run Code

In [2]:
```
import numpy as np
import tensorflow as tf
import matplotlib.pyplot as plt

print(tf.__version__)
```

```
2.0.0
```

In [3]:
```
# actual weight = 2 and actual bias = 0.9
x = np.linspace(0, 3, 120)
y = 2 * x + 0.9 + np.random.randn(*x.shape) * 0.3

plt.scatter(x,y, label="input data set")
```

Out[3]:
```
<matplotlib.collections.PathCollection at 0x7fbd9e5465d0>
```



Another Example with TensorFlow version 2 cont...

jupyter TensorflowIntro_v02_01 Last Checkpoint: Last Sunday at 10:28 (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

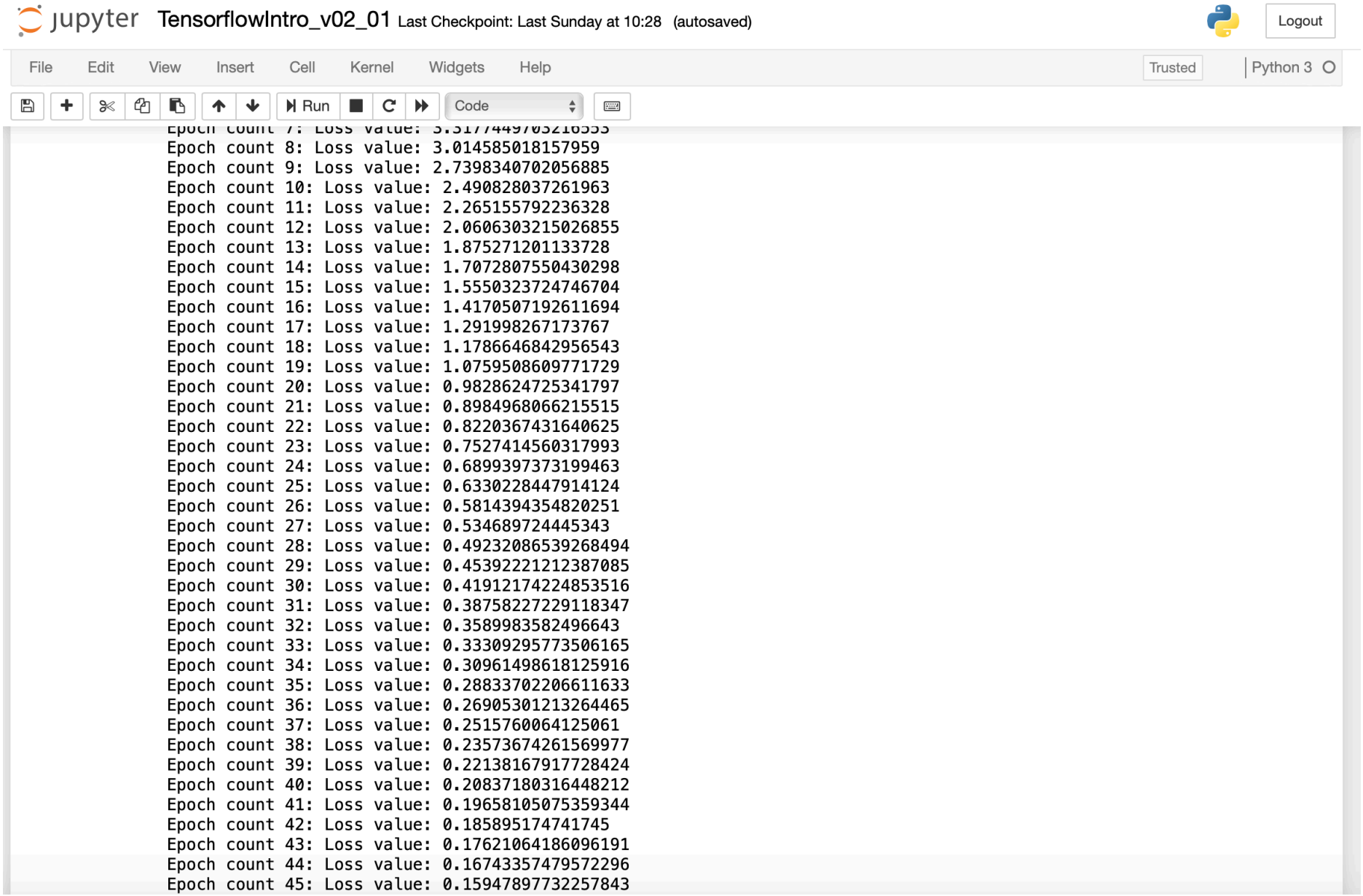
Run Code

In [4]:
```
class LinearModel:
    def __call__(self, x):
        return self.Weight * x + self.Bias

    def __init__(self):
        self.Weight = tf.Variable(11.0)
        self.Bias = tf.Variable(12.0)

def loss(y, pred):
    return tf.reduce_mean(tf.square(y - pred))

def train(linear_model, x, y, lr=0.12):
    with tf.GradientTape() as t:
        current_loss = loss(y, linear_model(x))

    lr_weight, lr_bias = t.gradient(current_loss, [linear_model.Weight, linear_model.Bias])
    linear_model.Weight.assign_sub(lr * lr_weight)
    linear_model.Bias.assign_sub(lr * lr_bias)
```

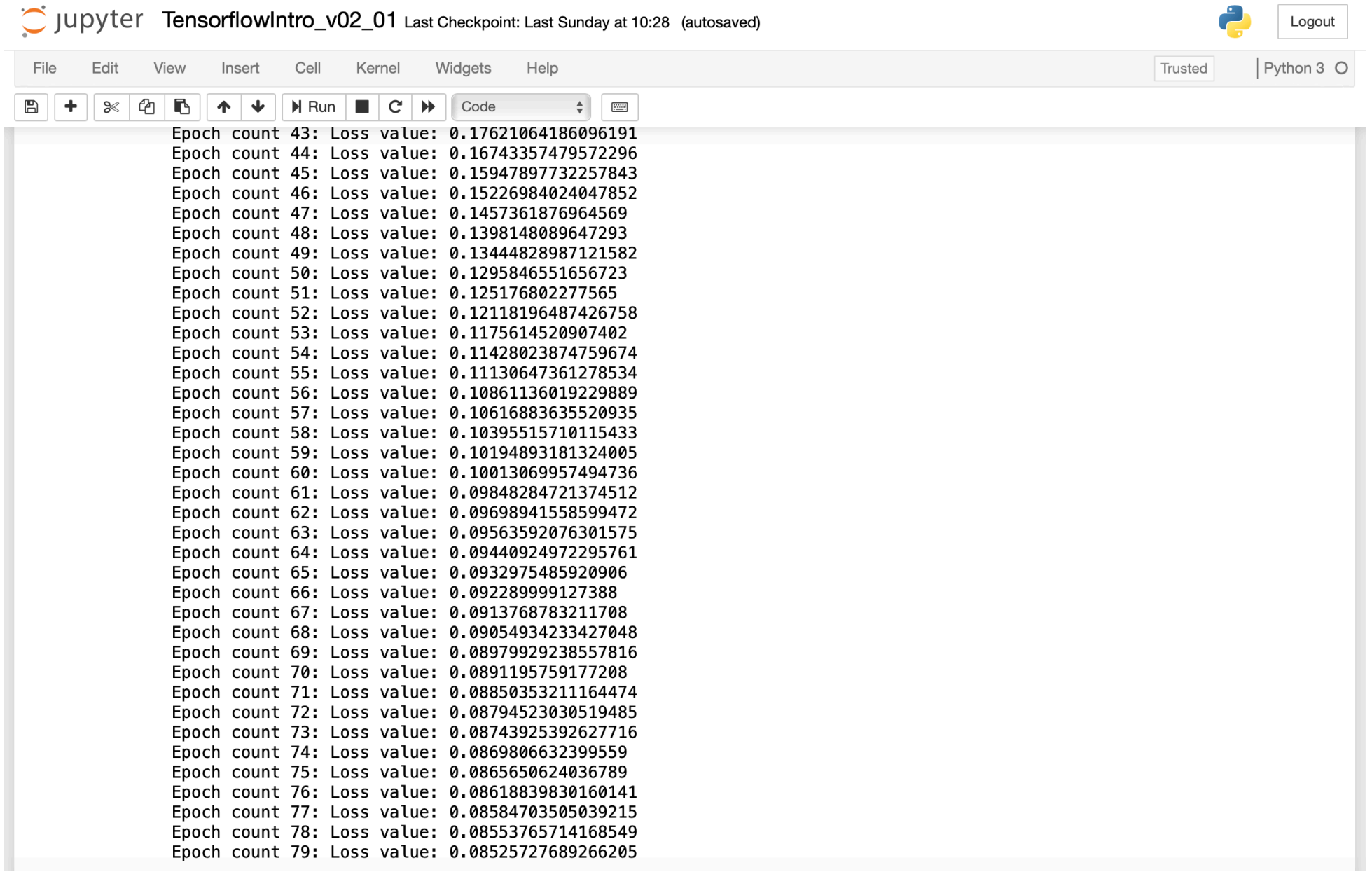
In [5]:
```
linear_model = LinearModel()
Weights, Biases = [], []
epochs = 80
for epoch_count in range(epochs):
    Weights.append(linear_model.Weight.numpy())
    Biases.append(linear_model.Bias.numpy())
    real_loss = loss(y, linear_model(x))
    train(linear_model, x, y, lr=0.12)
    print(f"Epoch count {epoch_count}: Loss value: {real_loss.numpy()}")
```

```
Epoch count 0: Loss value: 666.9447021484375
Epoch count 1: Loss value: 10.690195083618164
Epoch count 2: Loss value: 5.408223628997803
Epoch count 3: Loss value: 4.878205299377441
Epoch count 4: Loss value: 4.428596496582031
Epoch count 5: Loss value: 4.021341323852539
Epoch count 6: Loss value: 3.652250051498413
Epoch count 7: Loss value: 3.3177449703216553
```

Another Example with TensorFlow version 2 cont...



Another Example with TensorFlow version 2 cont...



Another Example with TensorFlow version 2 cont...

jupyter TensorflowIntro_v02_01 Last Checkpoint: Last Sunday at 10:28 (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

Run Code

```
Epoch count 61: Loss value: 0.09848284721374512
Epoch count 62: Loss value: 0.09698941558599472
Epoch count 63: Loss value: 0.09563592076301575
Epoch count 64: Loss value: 0.09440924972295761
Epoch count 65: Loss value: 0.0932975485920906
Epoch count 66: Loss value: 0.092289999127388
Epoch count 67: Loss value: 0.0913768783211708
Epoch count 68: Loss value: 0.09054934233427048
Epoch count 69: Loss value: 0.08979929238557816
Epoch count 70: Loss value: 0.0891195759177208
Epoch count 71: Loss value: 0.08850353211164474
Epoch count 72: Loss value: 0.08794523030519485
Epoch count 73: Loss value: 0.08743925392627716
Epoch count 74: Loss value: 0.0869806632399559
Epoch count 75: Loss value: 0.0865650624036789
Epoch count 76: Loss value: 0.08618839830160141
Epoch count 77: Loss value: 0.08584703505039215
Epoch count 78: Loss value: 0.08553765714168549
Epoch count 79: Loss value: 0.08525727689266205
```

In [6]:
```
linear_model.Weight.numpy(), linear_model.Bias.numpy()
```

Out[6]: (2.0053532, 0.89694154)

In [7]:
```
RMSE = loss(y, linear_model(x))

RMSE.numpy()
```

Out[7]: 0.08500317

In [ ]: