Deep Learning Using TensorFlow

Dr. Ash Pahwa

Lesson 5: Linear Regression in TensorFlow

Lesson 5.1: Linear Regression in TensorFlow

2 variables

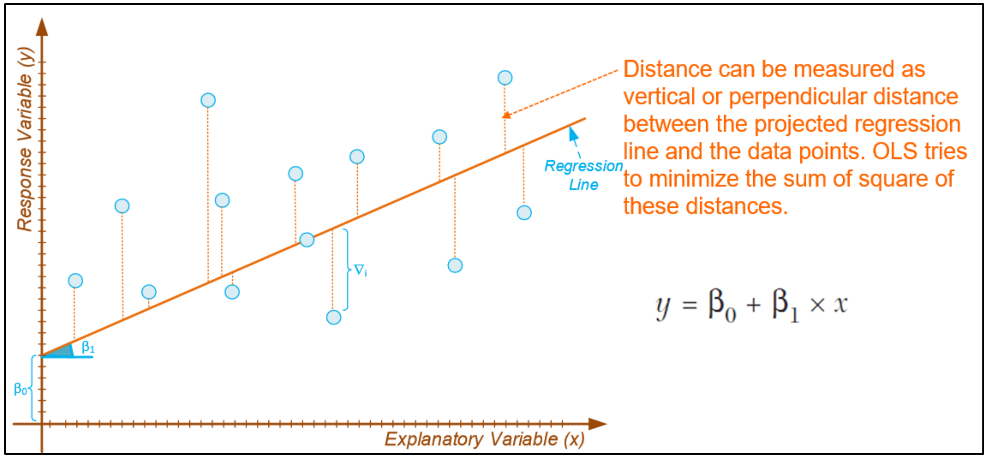
Outline

- Linear Regression
- Linear Regression in Scikit-Learn
- Linear Regression in TensorFlow

Linear Regression

Computing the Regression Line Compute: Intercept and Slope

- Residual = Observed value – Computed Value
- Suppose regression equation is
  - $y = mx + b$
  - y *is the explanatory variable*
  - x *is the pedictor variable*
  - m *is the slope of the line*
  - b *is the intercept*
- $Residual = \ y_i - (mx_i + b)$
- $Residual^2 = \ (y_i - (mx_i + b))^2$
- $Residuals\ Sum\ of\ Squares = (RSS) = \sum_{i=1}^{N}(y_i - (mx_i + b))^2$



Partial Derivatives of the RSS w.r.t. Intercept and Slope

- $Residuals\ Sum\ of\ Squares = (RSS) = \sum_{i=1}^{N}(y_i - (mx_i + b))^2$
- To find the minimum point of this function,
  - we will take the partial derivative of RSS with respect to 'm' and 'b' and set that to zero.

- $RSS = \sum_{i=1}^{N}(y_i - (mx_i + b))^2$
- $\frac{\partial RSS(m,b)}{\partial b} = \sum_{i=1}^{N} \frac{\partial}{\partial b}(y_i - (mx_i + b))^2$
- $\frac{\partial RSS(m,b)}{\partial b} = -2 \sum_{i=1}^{N}(y_i - (mx_i + b))$

- $RSS = \sum_{i=1}^{N}(y_i - (mx_i + b))^2$
- $\frac{\partial RSS(m,b)}{\partial m} = \sum_{i=1}^{N} \frac{\partial}{\partial m}(y_i - (mx_i + b))^2$
- $\frac{\partial RSS(m,b)}{\partial m} = -2 \sum_{i=1}^{N}(y_i - (mx_i + b))x_i$

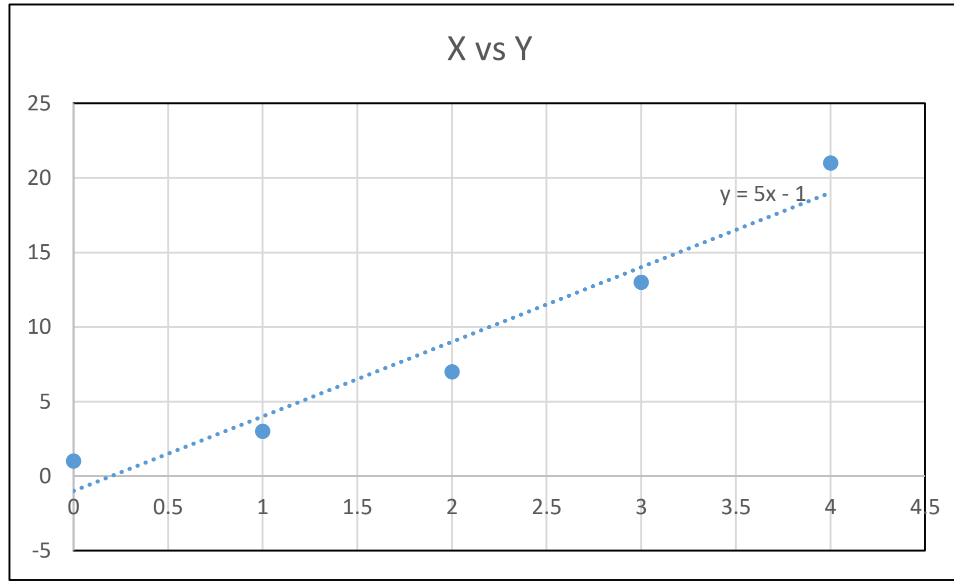
$$\nabla RSS(b,m) = \begin{vmatrix} \frac{\partial RSS(m,b)}{\partial b} \\ \frac{\partial RSS(m,b)}{\partial m} \end{vmatrix} = \begin{vmatrix} -2\sum_{i=1}^{N}(y_i - (mx_i + b)) \\ -2\sum_{i=1}^{N}(y_i - (mx_i + b))x_i \end{vmatrix} = 0$$

Gradient
Vector of Partial Derivatives

- To Compute 'm' and 'b'
  - SET GRADIENT = 0
- $\nabla\ RSS(b,m) = \begin{vmatrix} \frac{\partial RSS(m,b)}{\partial b} \\ \frac{\partial RSS(m,b)}{\partial m} \end{vmatrix} = \begin{vmatrix} -2\sum_{i=1}^{N}(y_i - (mx_i + b)) \\ -2\sum_{i=1}^{N}(y_i - (mx_i + b))x_i \end{vmatrix} = 0$
- ----------------------------------
- Top term
- $b = \left(\frac{\sum y_i}{N} - m\frac{\sum x_i}{N}\right) = \mu_y - m\mu_x$
- -------------------------------------
- Bottom term
- $m = \frac{\sum y_i x_i - \frac{\sum y_i \sum x_i}{N}}{\sum x_i^2 - \frac{\sum x_i \sum x_i}{N}} = r\frac{\sigma_y}{\sigma_x} = Correlation\,\frac{Std\ Dev\ of\ y}{Std\ Dev\ of\ x}$
- --

Example
Sample Data

	A	B	C
1			
2			
3		X	Y
4		0	1
5		1	3
6		2	7
7		3	13
8		4	21
9			



Basic Computations

$$m = \frac{\sum y_i x_i - \frac{\sum y_i \sum x_i}{N}}{\sum x_i^2 - \frac{\sum x_i \sum x_i}{N}}$$

$$b = \left(\frac{\sum y_i}{N} - m\frac{\sum x_i}{N}\right)$$

	A	B	C	D	E	F	G
1							
2							
3		X	Y		X*Y		X^2
4		0	1		0		0
5		1	3		3		1
6		2	7		14		4
7		3	13		39		9
8		4	21		84		16
9							
10	SUM	10	45		140		30
11	AVERAGE	2	9		28		6
12	StdDev	1.58113883	8.124038405				
13	Correlation	0.97312368					
14							

Method#1 Compute Slope

$$m = \frac{\sum y_i x_i - \frac{\sum y_i \sum x_i}{N}}{\sum x_i^2 - \frac{\sum x_i \sum x_i}{N}}$$

$$b = \left(\frac{\sum y_i}{N} - m\frac{\sum x_i}{N}\right)$$

	A	B	C	D	E	F	G
1							
2							
3		X	Y		X*Y		X^2
4		0	1		0		0
5		1	3		3		1
6		2	7		14		4
7		3	13		39		9
8		4	21		84		16
9							
10	SUM	10	45		140		30
11	AVERAGE	2	9		28		6
12	StdDev	1.58113883	8.124038405				
13	Correlation	0.97312368					
14							

$$m = \frac{\sum y_i x_i - \frac{\sum y_i \sum x_i}{N}}{\sum x_i^2 - \frac{\sum x_i \sum x_i}{N}} = \frac{140 - \frac{45 * 10}{5}}{30 - \frac{10 * 10}{5}} = \frac{50}{10} = 5$$

$$b = \left(\frac{\sum y_i}{N} - m\frac{\sum x_i}{N}\right) = \frac{45}{5} - 5 * \frac{10}{5} = -1$$

Regression Equation

$$y = 5x - 1$$

$$m = \frac{\sum y_i x_i - \frac{\sum y_i \sum x_i}{N}}{\sum x_i^2 - \frac{\sum x_i \sum x_i}{N}}$$

$$b = \left(\frac{\sum y_i}{N} - m\frac{\sum x_i}{N}\right)$$

Method#2
Compute Slope

	A	B	C	D	E	F	G
1							
2							
3		X	Y		X*Y		X^2
4		0	1		0		0
5		1	3		3		1
6		2	7		14		4
7		3	13		39		9
8		4	21		84		16
9							
10	SUM	10	45		140		30
11	AVERAGE	2	9		28		6
12	StdDev	1.58113883	8.124038405				
13	Correlation	0.97312368					
14							

- $$m = \frac{\sum y_i x_i - \frac{\sum y_i \sum x_i}{N}}{\sum x_i^2 - \frac{\sum x_i \sum x_i}{N}}$$
- Divide both numerator and denominator by N
- $$m = \frac{\frac{\sum y_i x_i}{N} - \frac{\sum y_i \sum x_i}{N.N}}{\frac{\sum x_i^2}{N} - \frac{\sum x_i \sum x_i}{N.N}} = \frac{Mean\ of\ X*Y - (Mean\ of\ X)*(Mean\ of\ Y)}{Mean\ of\ x^2 - (Mean\ of\ X)*(Mean\ of\ X)}$$

Regression Equation
$y = 5x - 1$

- $$m = \frac{Mean\ of\ X*Y - (Mean\ of\ X)*(Mean\ of\ Y)}{Mean\ of\ x^2 - (Mean\ of\ X)*(Mean\ of\ X)} = \frac{28-(2*9)}{6-(2*2)} = \frac{10}{2} = 5$$

$$b = \left(\frac{\sum y_i}{N} - m\frac{\sum x_i}{N}\right) = \frac{45}{5} - 5*\frac{10}{5} = -1$$

$$m = \frac{\sum y_i x_i - \frac{\sum y_i \sum x_i}{N}}{\sum x_i^2 - \frac{\sum x_i \sum x_i}{N}}$$

$$b = \left(\frac{\sum y_i}{N} - m\frac{\sum x_i}{N}\right)$$

Method#3 Compute Slope

- $m = \frac{\sum y_i x_i - \frac{\sum y_i \sum x_i}{N}}{\sum x_i^2 - \frac{\sum x_i \sum x_i}{N}}$
- $m = r\frac{\sigma_y}{\sigma_x} = Correlation \frac{Std\ Dev\ of\ y}{Std\ Dev\ of\ x}$

	A	B	C	D	E	F	G
1							
2							
3		X	Y		X*Y		X^2
4		0	1		0		0
5		1	3		3		1
6		2	7		14		4
7		3	13		39		9
8		4	21		84		16
9							
10	SUM	10	45		140		30
11	AVERAGE	2	9		28		6
12	StdDev	1.58113883	8.124038405				
13	Correlation	0.97312368					
14							

Regression Equation
$y = 5x - 1$

C29 | f_x | =B13*C12/B12

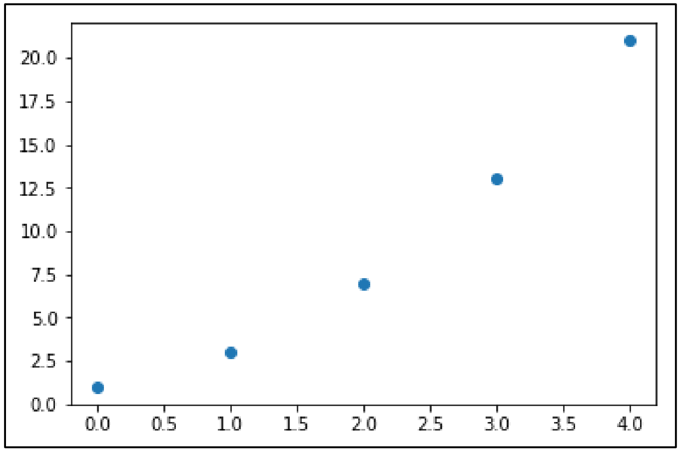
	A	B	C	D	E	F	G
27							
28							
29	Statistics	Slope	5		(Correlation * StdDev of Y) / StdDev of X		
30		Intercept	-1		(Mean of Y) - slope * (Mean of X)		
31							

Linear Regression in Scikit-Learn

Implementation in Scikit-Learn

X	Y
0	1
1	3
2	7
3	13
4	21

```
############################################################
# 1. Load the libraries
#
import numpy as np
import matplotlib.pyplot as plt
from sklearn import linear_model
#############################################
# Generate data
#
x = np.array([[0],[1],[2],[3],[4]])
y = np.array([[1],[3],[7],[13],[21]])

##################################################
# Linear Regression Using SKLearn function
#
linreg = linear_model.LinearRegression()
linreg.fit(x, y)
Out[16]: LinearRegression(copy_X=True, fit_intercept=True, n_jobs=1, normalize=False)
print (linreg.intercept_)
[-1.]
print (linreg.coef_)
[[ 5.]]
```



Regression Equation

$$y = 5x - 1$$

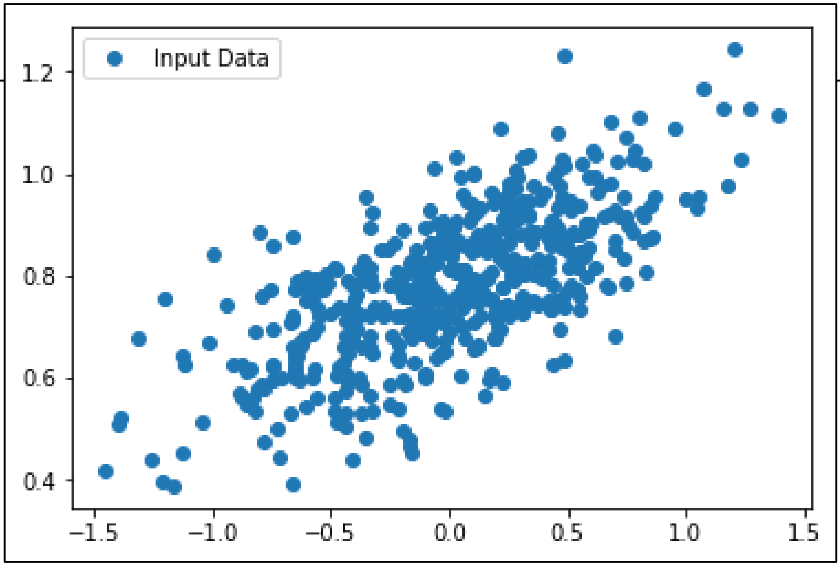
Example#2

Load the Libraries

```
import numpy as np

import matplotlib.pyplot as plt

import tensorflow as tf

from sklearn import linear_model

RANDOM_SEED = 42

tf.set_random_seed(RANDOM_SEED)
```

Generate the Dataset

```
###########################################
# Generate data
#
number_of_points = 500
x_point = []
y_point = []
m = 0.22
c = 0.78
for i in range(number_of_points):
    x = np.random.normal (0.0, 0.5)
    y = m*x + c + np.random.normal(0.0,0.1)
    x_point.append([x])
    y_point.append([y])



plt.plot(x_point, y_point, 'o', label='Input Data')
plt.legend()
Out[22]: <matplotlib.legend.Legend at 0x1a8a68333c8>
```



Linear Regression in Scikit-Learn

```
#################################################
# Linear Regression Using SKLearn function
#

linreg = linear_model.LinearRegression()

linreg.fit(x_point, y_point)
Out[27]: LinearRegression(copy_X=True, fit_intercept=True, n_jobs=1, normalize=False)

print (linreg.intercept_)
[ 0.78390055]

print (linreg.coef_)
[[ 0.20174445]]
```

Scikit-Learn: Answer

```
print (linreg.intercept_)
[ 0.78390055]

print (linreg.coef_)
[[ 0.20174445]]
```

Regression Equation

$$y = 0.2017x + 0.7839$$

Linear Regression in TensorFlow

Example#2

Build the TensorFlow Graph

```
graph = tf.Graph()
with graph.as_default():
    slope = tf.Variable(tf.random_uniform([1], -1.0, 1.0))
    intercept = tf.Variable(tf.zeros([1]))
    response = slope*x_point + intercept

    #############################################
    # Define Cost + Optimization Functions
    cost = tf.reduce_mean(tf.square(response - y_point))
    optimizer = tf.train.GradientDescentOptimizer(0.5).minimize(cost)
```

- $Residual = \; y_i - (mx_i + b)$
- $Residual^2 = \; (y_i - (mx_i + b))^2$
- $Residuals\ Sum\ of\ Squares = (RSS) = \; \sum_{i=1}^{N}(y_i - (mx_i + b))^2$

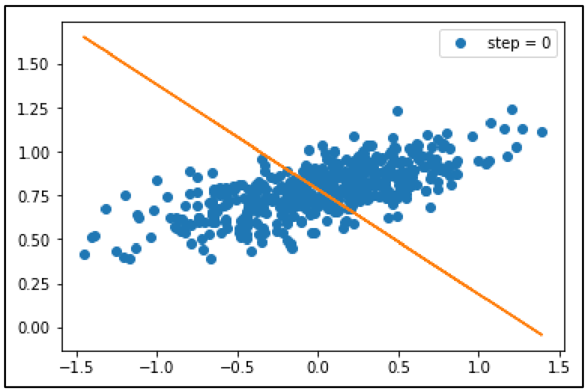
Linear Regression in TensorFlow

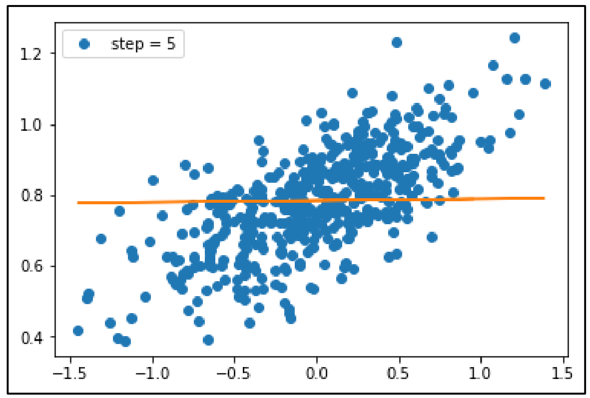
```
with tf.Session(graph=graph) as session:
    init = tf.global_variables_initializer()
    session.run(init)
    for epoch in range(30):
        session.run(optimizer)

        if ( epoch % 5 ) == 0:
            plt.plot(x_point, y_point, 'o', label = 'step = {}'.format(epoch))
            plt.plot(x_point,  session.run(slope)*x_point + session.run(intercept))
            plt.legend()
            plt.show()

    print("Slope = ",session.run(slope))
    print("Intercept = ",session.run(intercept))
```

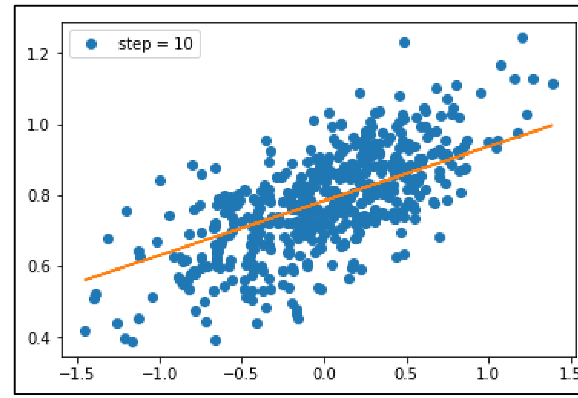
Result

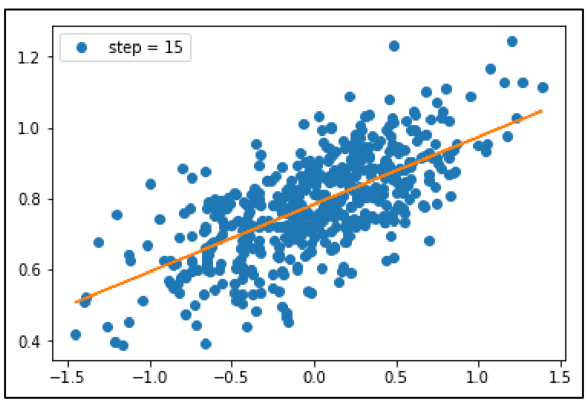
Scikit-Learn: Answer

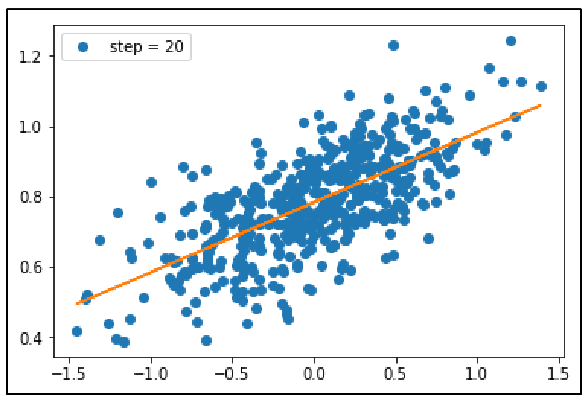
```
print (linreg.intercept_)
[ 0.78390055]

print (linreg.coef_)
[[ 0.20174445]]
```











TensorFlow Answer

```
Slope =   [ 0.20152435]
Intercept =   [ 0.78390092]
```

Regression Equation

$$y = 0.2015x + 0.7839$$

Summary

- Linear Regression
- Linear Regression in Scikit-Learn
- Linear Regression in TensorFlow

