UC San Diego Extension
Deep Learning Using TensorFlow
Winter 2019
Homework#6

Date Given: Feb 19, 2019 Due Date: Feb 25, 2019

==========

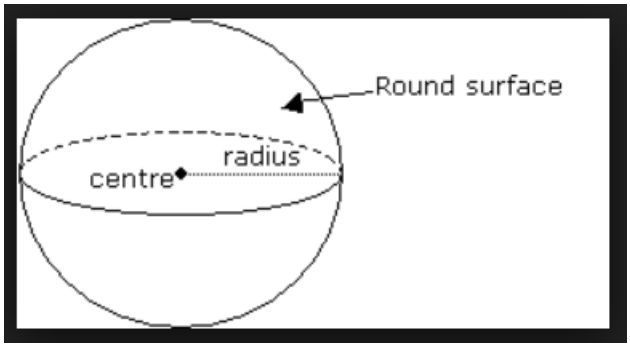
Problem#1

Suppose your cost function is as follows.

$$f(x,y) = z = -\sqrt{25 - (x-2)^2 - (y-3)^2}$$

This cost function represents the **bottom half** of a sphere with the following specifications.
Center of the sphere = x=2, y=3, z=0
Radius of the sphere = 5

The cost function is **convex** which means that it has a minimum point.



Find the values of 'x' and 'y' where the cost (z-value) is minimum using the Gradient Descent algorithm written in Python. The gradient (vector of partial derivatives) of this cost function is as follows.

$$\frac{\partial z}{\partial x} = \frac{2(x-2)}{\sqrt{25 - (x-2)^2 - (y-3)^2}}$$

$$\frac{\partial z}{\partial y} = \frac{2(y-3)}{\sqrt{25 - (x-2)^2 - (y-3)^2}}$$

Assuming learning rate = 0.01.

How many steps are needed for the Gradient Descent algorithm to converge? The criterion for convergence is when the incremental value of 'x' and 'y' is less than epsilon (0.000001).

Answer: The values of 'x' and 'y' where 'z' is minimum are as follows.
x = approximately 2, y = approximately 3. At this point the z = -5.

==========

Problem#2

In Homework#5/Problem#1, you computed the regression equation of the following data.

Dataset:

Generate a synthetic dataset using the following Python code.

```
-----------------------------------------------------------------
RANDOM_SEED = 42
tf.set_random_seed(RANDOM_SEED)

import numpy as np
import matplotlib.pyplot as plt

n_samples = 30
train_x = np.linspace(0,20,n_samples)
train_y = 3.7 * train_x + 14 + 4 * np.random.randn(n_samples)

plt.plot(train_x, train_y,'o')
-----------------------------------------------------------------
```

The value of RANDOM_SEED can be any integer. However, choose the value of RANDOM_SEED as 42. This will allow all students to get identical datasets. The 'train_x' is the predictor variable and 'train_y' is the response variable.

Model Building:

Compute the regression equation between 'train_x' and 'train_y' variables using Scikit-Learn package.

Gradient Descent (GD):

Using the Gradient Descent algorithm compute the regression equation - coefficient and intercept values.

Do not use the TensorFlow Gradient Descent function but write your own Python code to implement the GD algorithm.

Vary the learning rate and the number of iterations, till the answer computed by your GD algorithm (implemented in Python) matches with the answer computed by Scikit-Learn.