COSC4337_110-Saving -and-Loading-Models

1 Saving and Loading Models

[1]:
```
import numpy as np
import tensorflow as tf
import matplotlib.pyplot as plt
%matplotlib inline
```

[2]:
```
np.random.seed(101)
tf.set_random_seed(101)
```

1.1 Full Network Example

Let's work on a regression example, we are trying to solve a very simple equation:

y = mx + b

y will be the y_labels and x is the x_data. We are trying to figure out the slope and the intercept for the line that best fits our data!

1.1.1 Artifical Data (Some Made Up Regression Data)

[3]:
```
x_data = np.linspace(0,10,10) + np.random.uniform(-1.5,1.5,10)
```

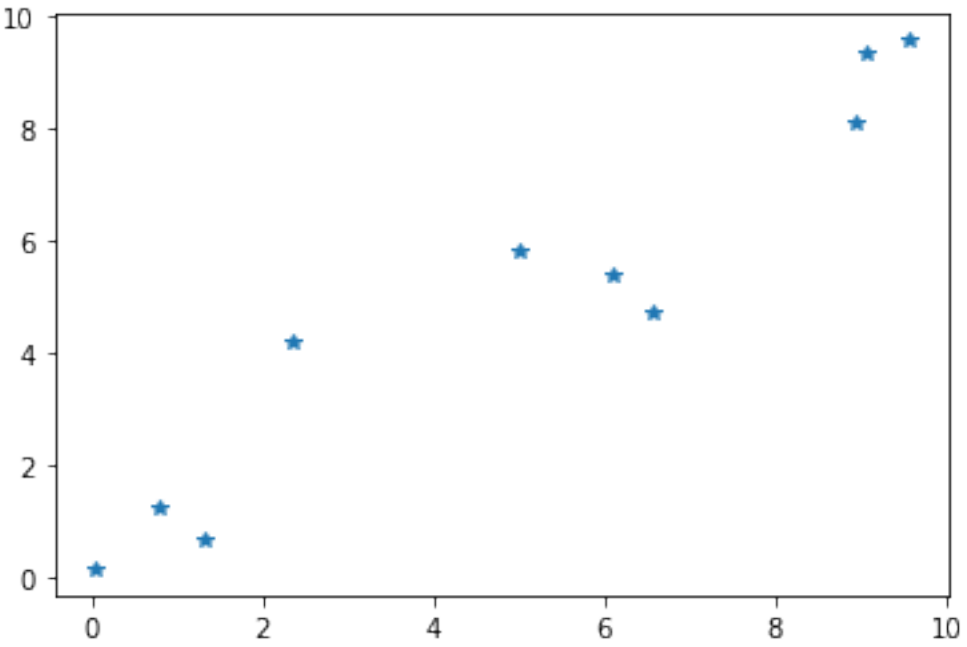
[4]:
```
x_data
```

[4]:
```
array([0.04919588, 1.32311387, 0.8076449 , 2.3478983 , 5.00027539,
       6.55724614, 6.08756533, 8.95861702, 9.55352047, 9.06981686])
```

[5]:
```
y_label = np.linspace(0,10,10) + np.random.uniform(-1.5,1.5,10)
```

[6]:
```
plt.plot(x_data,y_label,'*')
```

[6]:
```
[<matplotlib.lines.Line2D at 0x279c47eb048>]
```



** Variables **

[7]:
```
np.random.rand(2)
```

[7]:
```
array([0.68530633, 0.51786747])
```

[8]:
```
m = tf.Variable(0.39)
b = tf.Variable(0.2)
```

1.1.2 Cost Function

[9]:
```
error = tf.reduce_mean(y_label - (m*x_data+b))
```

1.1.3 Optimizer

[10]:
```
optimizer = tf.train.GradientDescentOptimizer(learning_rate=0.001)
train = optimizer.minimize(error)
```

1.1.4 Initialize Variables

[11]:
```
init = tf.global_variables_initializer()
```

2 Saving The Model

[12]:
```
saver = tf.train.Saver()
```

2.0.1 Create Session and Run!

[13]:
```
with tf.Session() as sess:

    sess.run(init)

    epochs = 100

    for i in range(epochs):

        sess.run(train)

    # Fetch Back Results
    final_slope , final_intercept = sess.run([m,b])

    # ONCE YOU ARE DONE
    # GO AHEAD AND SAVE IT!
    # Make sure to provide a directory for it to make or go to. May get errors␣
↪otherwise
    #saver.save(sess,'models/my_first_model.ckpt')
    saver.save(sess,'new_models/my_second_model.ckpt')
```

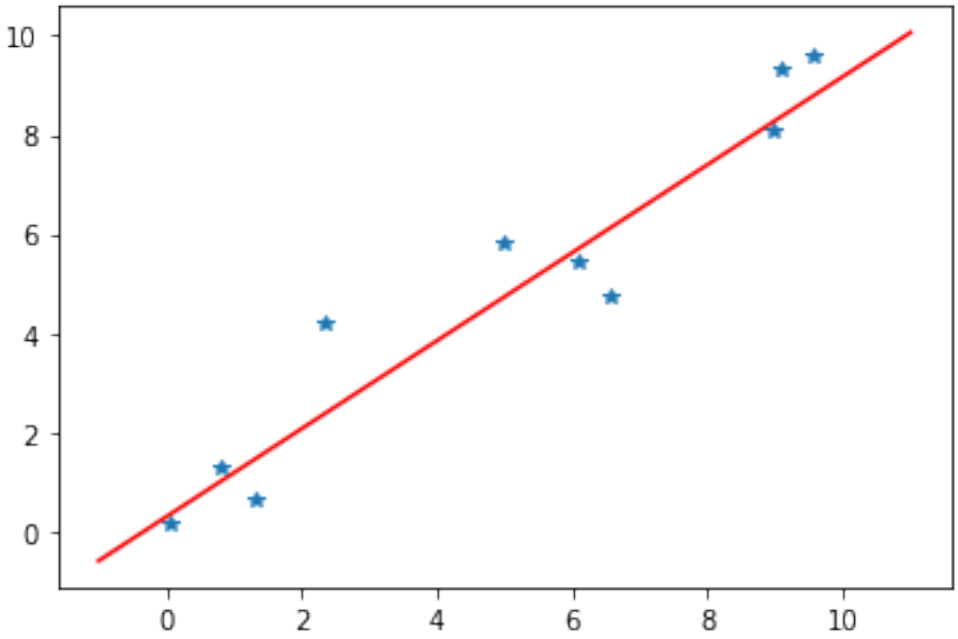
2.0.2 Evaluate Results

[14]:
```
x_test = np.linspace(-1,11,10)
y_pred_plot = final_slope*x_test + final_intercept

plt.plot(x_test,y_pred_plot,'r')

plt.plot(x_data,y_label,'*')
```

[14]:
```
[<matplotlib.lines.Line2D at 0x279c4976fc8>]
```



3 Loading a Model

[15]:
```
with tf.Session() as sess:

    # Restore the model
    saver.restore(sess,'new_models/my_second_model.ckpt')


    # Fetch Back Results
    restored_slope , restored_intercept = sess.run([m,b])
```

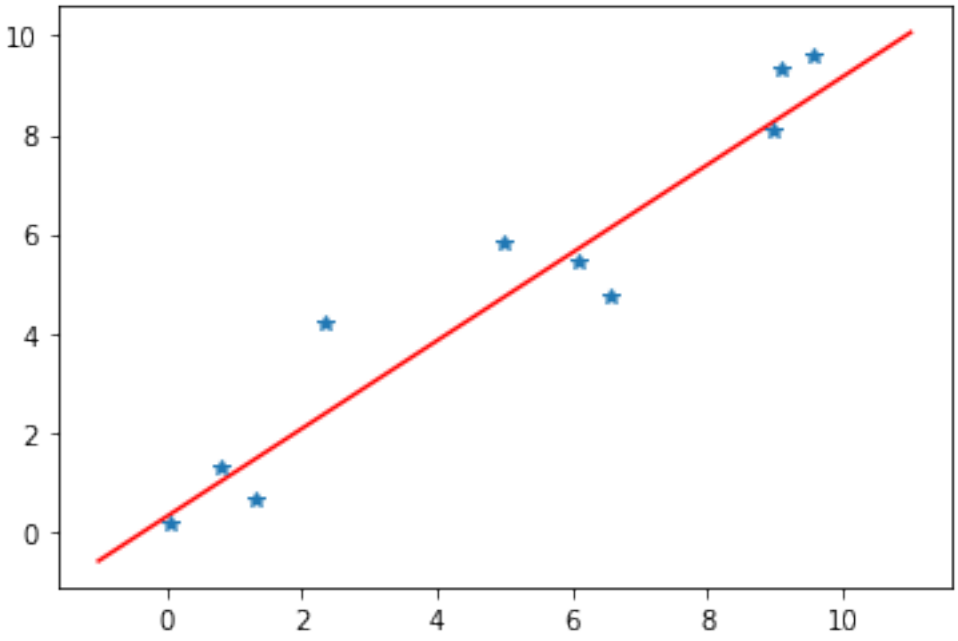
```
INFO:tensorflow:Restoring parameters from new_models/my_second_model.ckpt
```

[16]:
```
x_test = np.linspace(-1,11,10)
y_pred_plot = restored_slope*x_test + restored_intercept

plt.plot(x_test,y_pred_plot,'r')

plt.plot(x_data,y_label,'*')
```

[16]:
```
[<matplotlib.lines.Line2D at 0x279c4984b08>]
```


10
8
6
4
2
0
0
2
4
6
8
10

[ ]:

[ ]: