Question 4f

NB numpy had to be downgraded from 1.17.0 to 1.16.6 to remove an incompatibility with tensorflow 1.14, which manifests itself as a FutureWarning.

In [1]:

```
# Import Numpy and Tensorflow.

import numpy as np
import tensorflow as tf


# For reproducibility.

def reset_graph(seed=1):
    np.random.seed(seed)
    tf.reset_default_graph()
    tf.set_random_seed(seed)


# Import California housing data and StandardScaler from Scikit-Learn.

from sklearn.datasets import fetch_california_housing
from sklearn.preprocessing import StandardScaler

housing = fetch_california_housing()
m, n = housing.data.shape

scaler = StandardScaler()
scaled_housing_data = scaler.fit_transform(housing.data)
scaled_housing_data_plus_bias = np.c_[np.ones((m, 1)), scaled_housing_data]

housing_data_target = housing.target.reshape(-1, 1)


# Setup computational graph using placeholders.

reset_graph ()

learning_rate = 0.01

X = tf.placeholder(tf.float32, shape=(None, n + 1), name="X")
y = tf.placeholder(tf.float32, shape=(None, 1), name="y")

theta = tf.Variable(tf.random_uniform([n + 1, 1], -1.0, 1.0, seed=1), name="th
eta")
y_pred = tf.matmul(X, theta, name="predictions")
error = y_pred - y
mse = tf.reduce_mean(tf.square(error), name="mse")
optimizer = tf.train.GradientDescentOptimizer(learning_rate=learning_rate)
training_op = optimizer.minimize(mse)
```

```
# Define fetch_batch() for Mini-Batch Gradient Descent.

def fetch_batch(epoch, batch_index, batch_size):
    np.random.seed(epoch * n_batches + batch_index)
    indices = np.random.randint(m, size=batch_size)
    X_batch = scaled_housing_data_plus_bias[indices]
    y_batch = housing_data_target[indices]
    return X_batch, y_batch


# Execute computational graph using Mini-Batch Gradient Descent.

n_epochs = 10
batch_size = 100
n_batches = int(np.ceil(m / batch_size))

init = tf.global_variables_initializer()

with tf.Session() as sess:
    sess.run(init)
    for epoch in range(n_epochs):
        for batch_index in range(n_batches):
            X_batch, y_batch = fetch_batch(epoch, batch_index, batch_size)
            sess.run(training_op, feed_dict={X: X_batch, y: y_batch})
    best_theta = theta.eval()


# Output theta.

best_theta
```

Out[1]:

```
array([[ 2.0702078 ],
       [ 0.8462986 ],
       [ 0.12034925],
       [-0.27819806],
       [ 0.35567525],
       [ 0.00397728],
       [-0.01187481],
       [-0.86780626],
       [-0.8345916 ]], dtype=float32)
```

In [ ]:

In [ ]: