Q3:

1.
$$E[\frac{1}{m}\sum_{i\in I} a_i] = \frac{1}{m}\sum_{i\in I_0}^{I_m} E[a_i] = \frac{1}{m}\sum_{i\in I_0}^{I_m}(\sum_{j=1}^{n} P(i=j)a_i)$$
$$= \frac{1}{m}\sum_{i\in I_0}^{I_m}\frac{1}{n}(\sum_{j=1}^{n} a_i) = \frac{1}{m}\sum_{i\in I_0}^{I_m} E[a] = E[a] = \frac{1}{n}\sum_{i=1}^{n} a_i$$

2.
$$E[\nabla L(x,y,\theta)] = E[\frac{1}{m}\sum_{i=1}^{m}\nabla L(x_i,y_i,\theta)] = \frac{1}{m}\sum_{i=1}^{m}(\sum_{j=1}^{n} P(i=j)\nabla L(x_j,y_j,\theta))$$
$$= \frac{1}{m}\sum_{i=1}^{m}(\frac{1}{n}\sum_{j=1}^{n}\nabla L(x_j,y_j,\theta)) = \nabla L(x_j,y_j,\theta)$$

3. Write, in a sentence, the importance of this result.
It can give us a more accurate prediction on gradient. Because the mini-batch gradient estimator is not biased.

4. (a). Write down the gradient for a linear regression model with cost function.
$$\frac{1}{m}\sum_{i=0}^{m-1}(2X^T Xw - 2X^T y)$$

4. (b). Write code to compute this gradient.

```python
#TODO: implement linear regression gradient
def lin_reg_gradient(X, y, w):
    '''
    Compute gradient of linear regression model parameterized by w
    '''

    # the gradient of "(y - (w^T)X)^2" is 2((X.T)Xw - (X.T)y)

    # (X.T)X
    element1 = np.dot(X.T, X)

    # (X.T)y
    element2 = np.dot(X.T, y)

    #(X.T)Xw
    element3 = np.dot(element1, w)

    # 2((X.T)Xw - (X.T)y)
    element4 = 2 * (element3 - element2)

    # compute the gradient as result
    result = np.divide(element4, X.shape[0])

    '''
    a = y - X * w
    double = -2 * X.T
    scale = double * a
    return scale / X.shape[0]
    '''
    return result
```

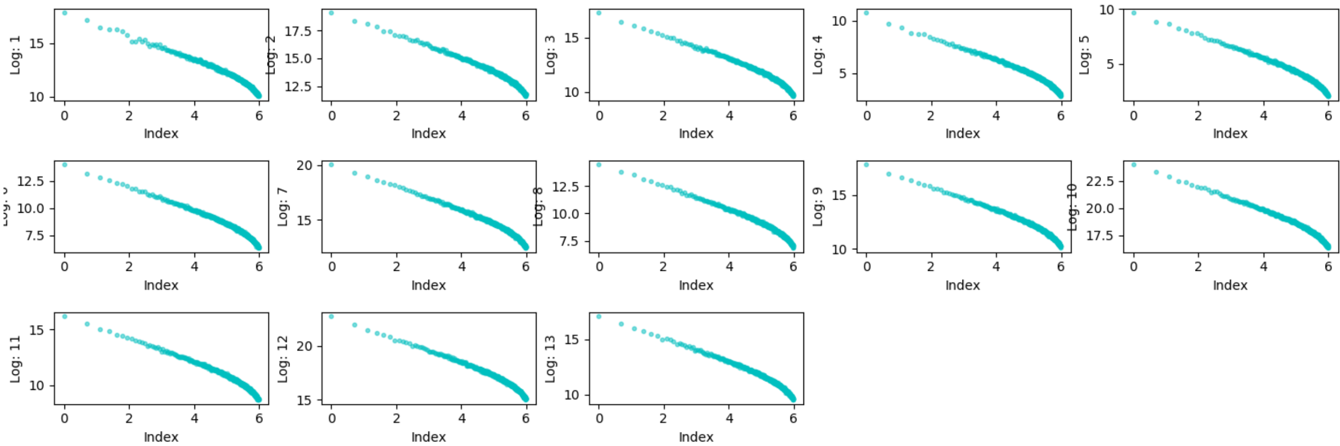
```python
# by assignment handout q3.5.
K = 500
m = 50

# Compute the actual gradient g.
g = lin_reg_gradient(X, y, w)

# Loop 500 time as assignment requirements to sum up all gradients in order to get gradients mean below.
total = 0
for i in range(K):
    X_b, y_b = batch_sampler.get_batch(m=m)
    total += lin_reg_gradient(X_b, y_b, w)

# Compute gradients mean.
ave_g = np.divide(total, K)
```

5. Randomly initialize the weight parameters for your model from a $N(0, I)$ distribution. Compare the value you have computed to the true gradient using both the squared distance metric and cosine similarity. Which is a more meaningful measure in this case and why?

```
------ Loading Boston Houses Dataset ------
Loaded...
Total data points: 506
Feature count: 13
Random parameters, w: [-1.22045083 -0.84157409 -1.27341559 -1.29841286 -0.6161908   0.22727756
 -0.33070065 -1.45357747  0.37455478 -1.96226453  1.40715624 -1.70147702
  0.81038213]
-------------------------------------------



Squared distance: 4199258.792648411
Cosine similarity: 0.9999997994324527
Accuracy: [ -1.60960978e-01  -6.80780234e-02  -5.91407318e-02  -1.02940144e+01
  -1.24905515e+00  -1.13327697e-01  -9.99030574e-03  -1.96106157e-01
  -6.48408268e-02  -1.61614333e-03  -3.80429362e-02  -1.98274271e-03
  -5.36222710e-02]

Process finished with exit code 0
```

Cosine similarity (cos) is more meaningful measure in this case. Because it can measure cohesion within clusters.

6.


Log: 1
Log: 2
Log: 3
Log: 4
Log: 5
Log: 7
Log: 8
Log: 9
Log: 10
Log: 11
Log: 12
Log: 13
Index