**you will build 6 models. You need to train Logistic Regression/Regularized Logistic Regression each with Batch Gradient Descent, Stochastic Gradient Descent and Mini Batch Gradient Descent. Also, you should plot their objective values versus epochs and compare their training and testing accuracy. You will need to tune the parameters a little bit to obtain reasonable results.**

**You do not have to follow the following procedure. You may implement your own functions and methods, but you need to show your results and plots.**

In [1]:
```
# Load Packages
import sklearn
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
```

# 1. Data processing

- Download the Breast Cancer dataset from canvas or from https://archive.ics.uci.edu/ml/datasets/breast+cancer+wisconsin+(diagnostic) (https://archive.ics.uci.edu/ml/datasets/breast+cancer+wisconsin+(diagnostic))
- Load the data.
- Preprocess the data.

# 1.1. Load the data

In [2]:
```
data = pd.read_csv("data.csv")
data
```

Out[2]:

| | id | diagnosis | radius_mean | texture_mean | perimeter_mean | area_mean | smoothness_ |
|---|---|---|---|---|---|---|---|
| **0** | 842302 | M | 17.99 | 10.38 | 122.80 | 1001.0 | 0 |
| **1** | 842517 | M | 20.57 | 17.77 | 132.90 | 1326.0 | 0 |
| **2** | 84300903 | M | 19.69 | 21.25 | 130.00 | 1203.0 | 0 |
| **3** | 84348301 | M | 11.42 | 20.38 | 77.58 | 386.1 | 0 |
| **4** | 84358402 | M | 20.29 | 14.34 | 135.10 | 1297.0 | 0 |
| **...** | ... | ... | ... | ... | ... | ... | |
| **564** | 926424 | M | 21.56 | 22.39 | 142.00 | 1479.0 | 0 |
| **565** | 926682 | M | 20.13 | 28.25 | 131.20 | 1261.0 | 0 |
| **566** | 926954 | M | 16.60 | 28.08 | 108.30 | 858.1 | 0 |
| **567** | 927241 | M | 20.60 | 29.33 | 140.10 | 1265.0 | 0 |
| **568** | 92751 | B | 7.76 | 24.54 | 47.92 | 181.0 | 0 |

569 rows × 33 columns

## 1.2 Examine and clean data

In [3]:
```
# Some columns may not be useful for the model (For example, the first column
# You need to get rid of the ID number feature.
# Also you should transform target labels in the second column from 'B' and 'M
# Dropped unnecessary columns
data.drop(data.columns[[0, 32]], axis = 1, inplace = True)

# Replacing M with 0 and B with 1
data["diagnosis"].replace({"M": -1, "B": 1}, inplace = True) # Replacing M wit
data
```

Out[3]:

| | diagnosis | radius_mean | texture_mean | perimeter_mean | area_mean | smoothness_mean | cor |
|---|---|---|---|---|---|---|---|
| **0** | -1 | 17.99 | 10.38 | 122.80 | 1001.0 | 0.11840 | |
| **1** | -1 | 20.57 | 17.77 | 132.90 | 1326.0 | 0.08474 | |
| **2** | -1 | 19.69 | 21.25 | 130.00 | 1203.0 | 0.10960 | |
| **3** | -1 | 11.42 | 20.38 | 77.58 | 386.1 | 0.14250 | |
| **4** | -1 | 20.29 | 14.34 | 135.10 | 1297.0 | 0.10030 | |
| **...** | ... | ... | ... | ... | ... | ... | |
| **564** | -1 | 21.56 | 22.39 | 142.00 | 1479.0 | 0.11100 | |
| **565** | -1 | 20.13 | 28.25 | 131.20 | 1261.0 | 0.09780 | |
| **566** | -1 | 16.60 | 28.08 | 108.30 | 858.1 | 0.08455 | |
| **567** | -1 | 20.60 | 29.33 | 140.10 | 1265.0 | 0.11780 | |
| **568** | 1 | 7.76 | 24.54 | 47.92 | 181.0 | 0.05263 | |

569 rows × 31 columns

## 1.3. Partition to training and testing sets

In [4]:
```
# You can partition using 80% training data and 20% testing data. It is a comm
# Dropping target feature and storing it in y.
X, y = np.asarray(data.iloc[:,1:]), np.asarray(data.iloc[:,0:1])
#splitting dataset in training and testing dataset.
x_train, x_test, y_train, y_test = train_test_split(X, y, test_size=0.20) # Te
```

## 1.4. Feature scaling

Use the standardization to transform both training and test features

In [5]:
```
# Standardization
import numpy

# calculate mu and sig using the training set
d = x_train.shape[1]
mu = numpy.mean(x_train, axis=0).reshape(1, d)
sig = numpy.std(x_train, axis=0).reshape(1, d)

# transform the training features
x_train = (x_train - mu) / (sig + 1E-6)

# transform the test features
x_test = (x_test - mu) / (sig + 1E-6)

print('test mean = ')
print(numpy.mean(x_test, axis=0))

print('test std = ')
print(numpy.std(x_test, axis=0))
```

```
test mean =
[ 0.09846829  0.02105125  0.09716162  0.13319684  0.04198499  0.04983166
  0.12638982  0.09525404  0.07707492  0.02220756  0.26450984  0.05625726
  0.27588685  0.35049629 -0.00287044 -0.01259705  0.05541236 -0.08305956
  0.2256218   0.05480612  0.11776221  0.05911909  0.12113957  0.1454764
  0.03303297  0.03909026  0.13357497  0.03778099  0.2038324   0.12162127]
test std =
[1.15561532 0.9916339  1.14949548 1.23696983 1.01375089 0.97291343
 1.08508832 1.068175   0.9237933  1.19951943 1.67120846 0.9575115
 1.73919545 2.22149969 1.11575893 0.96982156 1.20181084 0.94377501
 1.20810352 0.86159347 1.14110861 1.0073918  1.15753485 1.22125306
 1.08660755 1.11864561 1.1973734  1.02956169 1.1997496  1.3217767 ]
```

# 2. Logistic Regression Model

The objective function is $Q(w; X, y) = \frac{1}{n}\sum_{i=1}^{n}\log\left(1 + \exp\left(-y_i x_i^T w\right)\right) + \frac{\lambda}{2}\|w\|_2^2.$

When $\lambda = 0$, the model is a regular logistic regression and when $\lambda > 0$, it essentially becomes a regularized logistic regression.

In [6]:
```
# Calculate the objective function value, or loss
# Inputs:
#     w: weight: d-by-1 matrix
#     x: data: n-by-d matrix
#     y: label: n-by-1 matrix
#     lam: regularization parameter: scalar
# Return:
#     objective function value, or loss (scalar)
def objective(w, x, y, lam):

  yxw = np.dot(np.multiply(y,x),w)
  exponential_Term = np.exp(-yxw)

  Obj1 = np.mean(np.log(1 + exponential_Term))
  Obj2 = (lam / 2) * np.sum(w * w)

  objvalue = Obj1 + Obj2

  return objvalue

  pass
```

# 3. Numerical optimization

## 3.1. Gradient descent

The gradient at $w$ for regularized logistic regression is $g = -\frac{1}{n}\sum_{i=1}^{n}\frac{y_i x_i}{1+\exp\ (y_i x_i^T w)} + \lambda w$

In [7]:
```
# Calculate the gradient
# Inputs:
#     w: weight: d-by-1 matrix
#     x: data: n-by-d matrix
#     y: label: n-by-1 matrix
#     lam: regularization parameter: scalar
# Return:
#     g: gradient: d-by-1 matrix

def gradient(w, x, y, lam):

  yx = np.multiply(y, x)
  exponential_Term = np.exp(np.dot(yx, w))

  G1 = -np.mean(np.divide(yx, 1 + exponential_Term),axis = 0).reshape(d,1)
  G2 = lam * w

  g = G1 + G2

  return g

  pass
```

In [8]:
```
# Gradient descent for solving logistic regression
# You will need to do iterative processes (loops) to obtain optimal weights in

# Inputs:
#     x: data: n-by-d matrix
#     y: label: n-by-1 matrix
#     lam: scalar, the regularization parameter
#     learning_rate: scalar
#     w: weights: d-by-1 matrix, initialization of w
#     max_epoch: integer, the maximal epochs
# Return:
#     w: weights: d-by-1 matrix, the solution
#     objvals: a record of each epoch's objective value

def gradient_descent(x, y, lam, learning_rate, w, max_epoch=100):

  objvals = np.zeros(max_epoch) # making an array to store objective values

  for z in range(0,max_epoch) :

    current_Objective_Value = objective(w, x, y, lam)
    current_Gradient = gradient(w, x, y, lam)

    w -= learning_rate  * current_Gradient # updating weight

    objvals[z] = current_Objective_Value

    print(f'Iteration number: {z+1} Current Objective Value: {objvals[z]}')

  return w, objvals

  pass
```

Use gradient_descent function to obtain your optimal weights and a list of objective values over each epoch.

In [9]:
```
# Train logistic regression
# You should get the optimal weights and a list of objective values by using g
lam = 0
learning_rate = 1
w = np.zeros((d,1))
w_GD, obj_value_gd = gradient_descent(x_train, y_train, lam, learning_rate, w)
```

```
Iteration number: 1 Current Objective Value: 0.6931471805599453
Iteration number: 2 Current Objective Value: 0.17001750257758133
Iteration number: 3 Current Objective Value: 0.13895204379757714
Iteration number: 4 Current Objective Value: 0.12201292707159705
Iteration number: 5 Current Objective Value: 0.11308096191112656
Iteration number: 6 Current Objective Value: 0.10782741774777573
Iteration number: 7 Current Objective Value: 0.10389073243630481
Iteration number: 8 Current Objective Value: 0.1006187822611476
Iteration number: 9 Current Objective Value: 0.09778710533194178
Iteration number: 10 Current Objective Value: 0.09528563925883861
Iteration number: 11 Current Objective Value: 0.09304698843075072
Iteration number: 12 Current Objective Value: 0.09102441960127003
Iteration number: 13 Current Objective Value: 0.08918319244407318
Iteration number: 14 Current Objective Value: 0.08749639350001463
Iteration number: 15 Current Objective Value: 0.08594260844205202
Iteration number: 16 Current Objective Value: 0.08450447334101523
Iteration number: 17 Current Objective Value: 0.08316770358024018
Iteration number: 18 Current Objective Value: 0.08192040910841912
Iteration number: 19 Current Objective Value: 0.08075259420997562
Iteration number: 20 Current Objective Value: 0.07965578255136083
Iteration number: 21 Current Objective Value: 0.0786227305688932
Iteration number: 22 Current Objective Value: 0.07764720493453865
Iteration number: 23 Current Objective Value: 0.07672380751915882
Iteration number: 24 Current Objective Value: 0.07584783617747447
Iteration number: 25 Current Objective Value: 0.07501517293799068
Iteration number: 26 Current Objective Value: 0.07422219341542996
Iteration number: 27 Current Objective Value: 0.07346569283344026
Iteration number: 28 Current Objective Value: 0.0727428251712
Iteration number: 29 Current Objective Value: 0.0720510527682928
Iteration number: 30 Current Objective Value: 0.07138810432902253
Iteration number: 31 Current Objective Value: 0.0707519397214567
Iteration number: 32 Current Objective Value: 0.07014072031001209
Iteration number: 33 Current Objective Value: 0.06955278382277881
Iteration number: 34 Current Objective Value: 0.06898662295694732
Iteration number: 35 Current Objective Value: 0.06844086708273224
Iteration number: 36 Current Objective Value: 0.06791426652906542
Iteration number: 37 Current Objective Value: 0.0674056790311564
Iteration number: 38 Current Objective Value: 0.0669140579968097
Iteration number: 39 Current Objective Value: 0.06643844230966715
Iteration number: 40 Current Objective Value: 0.06597794743672454
Iteration number: 41 Current Objective Value: 0.06553175764716104
Iteration number: 42 Current Objective Value: 0.06509911918171696
Iteration number: 43 Current Objective Value: 0.06467933423810221
Iteration number: 44 Current Objective Value: 0.0642717556594186
Iteration number: 45 Current Objective Value: 0.06387578223026744
Iteration number: 46 Current Objective Value: 0.06349085449983345
Iteration number: 47 Current Objective Value: 0.06311645106336482
Iteration number: 48 Current Objective Value: 0.06275208524357513
Iteration number: 49 Current Objective Value: 0.06239730212194293
Iteration number: 50 Current Objective Value: 0.06205167587697785
Iteration number: 51 Current Objective Value: 0.06171480739249536
Iteration number: 52 Current Objective Value: 0.06138632210399215
Iteration number: 53 Current Objective Value: 0.061065868055493445
Iteration number: 54 Current Objective Value: 0.06075311414288571
Iteration number: 55 Current Objective Value: 0.060447748522853686
Iteration number: 56 Current Objective Value: 0.06014947716919832
Iteration number: 57 Current Objective Value: 0.05985802256059315
```

```
Iteration number: 58 Current Objective Value: 0.059573122485797755
Iteration number: 59 Current Objective Value: 0.059294528954039895
Iteration number: 60 Current Objective Value: 0.059022007199741516
Iteration number: 61 Current Objective Value: 0.05875533477203237
Iteration number: 62 Current Objective Value: 0.058494300700598016
Iteration number: 63 Current Objective Value: 0.05823870473036931
Iteration number: 64 Current Objective Value: 0.05798835661839869
Iteration number: 65 Current Objective Value: 0.05774307548700278
Iteration number: 66 Current Objective Value: 0.05750268922789274
Iteration number: 67 Current Objective Value: 0.05726703395257987
Iteration number: 68 Current Objective Value: 0.05703595348484029
Iteration number: 69 Current Objective Value: 0.0568092988914617
Iteration number: 70 Current Objective Value: 0.05658692804788195
Iteration number: 71 Current Objective Value: 0.05636870523567262
Iteration number: 72 Current Objective Value: 0.05615450076912472
Iteration number: 73 Current Objective Value: 0.055944190648463225
Iteration number: 74 Current Objective Value: 0.05573765623745811
Iteration number: 75 Current Objective Value: 0.055534783963413166
Iteration number: 76 Current Objective Value: 0.05533546503770483
Iteration number: 77 Current Objective Value: 0.05513959519521464
Iteration number: 78 Current Objective Value: 0.05494707445115097
Iteration number: 79 Current Objective Value: 0.054757806873893526
Iteration number: 80 Current Objective Value: 0.054571700372616605
Iteration number: 81 Current Objective Value: 0.054388666498557885
Iteration number: 82 Current Objective Value: 0.05420862025889917
Iteration number: 83 Current Objective Value: 0.05403147994231497
Iteration number: 84 Current Objective Value: 0.05385716695532627
Iteration number: 85 Current Objective Value: 0.05368560566866932
Iteration number: 86 Current Objective Value: 0.05351672327295631
Iteration number: 87 Current Objective Value: 0.05335044964296401
Iteration number: 88 Current Objective Value: 0.053186717209941364
Iteration number: 89 Current Objective Value: 0.05302546084137612
Iteration number: 90 Current Objective Value: 0.05286661772770562
Iteration number: 91 Current Objective Value: 0.0527101272754976
Iteration number: 92 Current Objective Value: 0.052555931006663954
Iteration number: 93 Current Objective Value: 0.052403972463304666
Iteration number: 94 Current Objective Value: 0.05225419711780958
Iteration number: 95 Current Objective Value: 0.052106552287874384
Iteration number: 96 Current Objective Value: 0.051960987056112726
Iteration number: 97 Current Objective Value: 0.051817452193970215
Iteration number: 98 Current Objective Value: 0.05167590008966776
Iteration number: 99 Current Objective Value: 0.05153628467992134
Iteration number: 100 Current Objective Value: 0.05139856138520395
```

In [10]:
```
# Train regularized logistic regression
# You should get the optimal weights and a list of objective values by using g
lam = 0.000001
learning_rate = 1
w = np.zeros((d,1))
w_GD_REG, obj_value_gd_reg = gradient_descent(x_train, y_train, lam, learning_
```

```
Iteration number: 1 Current Objective Value: 0.6931471805599453
Iteration number: 2 Current Objective Value: 0.1700185488924667
Iteration number: 3 Current Objective Value: 0.13895316356566675
Iteration number: 4 Current Objective Value: 0.1220141862382821
Iteration number: 5 Current Objective Value: 0.1130823805034201
Iteration number: 6 Current Objective Value: 0.10782898054663523
Iteration number: 7 Current Objective Value: 0.10389242655545354
Iteration number: 8 Current Objective Value: 0.10062059921686813
Iteration number: 9 Current Objective Value: 0.0977890387185298
Iteration number: 10 Current Objective Value: 0.0952876838959905
Iteration number: 11 Current Objective Value: 0.0930491399626965
Iteration number: 12 Current Objective Value: 0.09102667427034235
Iteration number: 13 Current Objective Value: 0.08918554694920747
Iteration number: 14 Current Objective Value: 0.08749884490211708
Iteration number: 15 Current Objective Value: 0.08594515409728437
Iteration number: 16 Current Objective Value: 0.08450711085169114
Iteration number: 17 Current Objective Value: 0.0831704307573864
Iteration number: 18 Current Objective Value: 0.08192322394239937
Iteration number: 19 Current Objective Value: 0.08075549484690218
Iteration number: 20 Current Objective Value: 0.07965876727379624
Iteration number: 21 Current Objective Value: 0.07862579777982176
Iteration number: 22 Current Objective Value: 0.0776503531438881
Iteration number: 23 Current Objective Value: 0.07672703533234955
Iteration number: 24 Current Objective Value: 0.07585114228560712
Iteration number: 25 Current Objective Value: 0.07501855610937554
Iteration number: 26 Current Objective Value: 0.0742256524882274
Iteration number: 27 Current Objective Value: 0.07346922670923067
Iteration number: 28 Current Objective Value: 0.07274643280933707
Iteration number: 29 Current Objective Value: 0.07205473318092258
Iteration number: 30 Current Objective Value: 0.071391856576671
Iteration number: 31 Current Objective Value: 0.07075576290910643
Iteration number: 32 Current Objective Value: 0.07014461358360108
Iteration number: 33 Current Objective Value: 0.06955674636606599
Iteration number: 34 Current Objective Value: 0.0689906539886977
Iteration number: 35 Current Objective Value: 0.06844496585418146
Iteration number: 36 Current Objective Value: 0.06791843232162996
Iteration number: 37 Current Objective Value: 0.06740991115436024
Iteration number: 38 Current Objective Value: 0.06691835578640218
Iteration number: 39 Current Objective Value: 0.06644280512591072
Iteration number: 40 Current Objective Value: 0.06598237466283316
Iteration number: 41 Current Objective Value: 0.06553624868787347
Iteration number: 42 Current Objective Value: 0.06510367346198999
Iteration number: 43 Current Objective Value: 0.06468395120191216
Iteration number: 44 Current Objective Value: 0.06427643476865934
Iteration number: 45 Current Objective Value: 0.06388052296373603
Iteration number: 46 Current Objective Value: 0.06349565635229425
Iteration number: 47 Current Objective Value: 0.06312131354468507
Iteration number: 48 Current Objective Value: 0.06275700787792483
Iteration number: 49 Current Objective Value: 0.06240228444705355
Iteration number: 50 Current Objective Value: 0.06205671744345408
Iteration number: 51 Current Objective Value: 0.06171990776317574
Iteration number: 52 Current Objective Value: 0.06139148085335377
Iteration number: 53 Current Objective Value: 0.06107108476909729
Iteration number: 54 Current Objective Value: 0.06075838841685916
Iteration number: 55 Current Objective Value: 0.0604530799634069
Iteration number: 56 Current Objective Value: 0.06015486539217205
Iteration number: 57 Current Objective Value: 0.059863467191035184
```

```
Iteration number: 58 Current Objective Value: 0.05957862315756595
Iteration number: 59 Current Objective Value: 0.05930008530942957
Iteration number: 60 Current Objective Value: 0.05902761888913536
Iteration number: 61 Current Objective Value: 0.05876100145357129
Iteration number: 62 Current Objective Value: 0.05850002203987127
Iteration number: 63 Current Objective Value: 0.0582444804001224
Iteration number: 64 Current Objective Value: 0.057994186298258116
Iteration number: 65 Current Objective Value: 0.057748958863215884
Iteration number: 66 Current Objective Value: 0.057508625993082145
Iteration number: 67 Current Objective Value: 0.057273023805511095
Iteration number: 68 Current Objective Value: 0.05704199613020187
Iteration number: 69 Current Objective Value: 0.056815394039656915
Iteration number: 70 Current Objective Value: 0.056593075414831225
Iteration number: 71 Current Objective Value: 0.056374904542626246
Iteration number: 72 Current Objective Value: 0.05616075174248478
Iteration number: 73 Current Objective Value: 0.055950493019614585
Iteration number: 74 Current Objective Value: 0.05574400974260756
Iteration number: 75 Current Objective Value: 0.05554118834343636
Iteration number: 76 Current Objective Value: 0.05534192003800051
Iteration number: 77 Current Objective Value: 0.055146100565565674
Iteration number: 78 Current Objective Value: 0.054953629945591845
Iteration number: 79 Current Objective Value: 0.05476441225058389
Iteration number: 80 Current Objective Value: 0.054578355393720454
Iteration number: 81 Current Objective Value: 0.054395370930128174
Iteration number: 82 Current Objective Value: 0.054215373870767344
Iteration number: 83 Current Objective Value: 0.054038282507985404
Iteration number: 84 Current Objective Value: 0.053864018251875076
Iteration number: 85 Current Objective Value: 0.05369250547664749
Iteration number: 86 Current Objective Value: 0.05352367137629676
Iteration number: 87 Current Objective Value: 0.05335744582889251
Iteration number: 88 Current Objective Value: 0.05319376126889099
Iteration number: 89 Current Objective Value: 0.053032552566905096
Iteration number: 90 Current Objective Value: 0.05287375691641845
Iteration number: 91 Current Objective Value: 0.05271731372696911
Iteration number: 92 Current Objective Value: 0.052563164523366374
Iteration number: 93 Current Objective Value: 0.05241125285053727
Iteration number: 94 Current Objective Value: 0.05226152418363114
Iteration number: 95 Current Objective Value: 0.05211392584303794
Iteration number: 96 Current Objective Value: 0.051968406914002736
Iteration number: 97 Current Objective Value: 0.051824918170542
Iteration number: 98 Current Objective Value: 0.05168341200338897
Iteration number: 99 Current Objective Value: 0.05154384235171557
Iteration number: 100 Current Objective Value: 0.05140616463839617
```

## 3.2. Stochastic gradient descent (SGD)

Define new objective function $Q_i(w) = \log\left(1 + \exp\left(-y_i x_i^T w\right)\right) + \frac{\lambda}{2}\|w\|_2^2$.

The stochastic gradient at $w$ is $g_i = \frac{\partial Q_i}{\partial w} = -\frac{y_i x_i}{1+\exp\left(y_i x_i^T w\right)} + \lambda w$.

You may need to implement a new function to calculate the new objective function and gradients.

In [11]:
```
# Calculate the objective Q_i and the gradient of Q_i
# Inputs:
#     w: weights: d-by-1 matrix
#     xi: data: 1-by-d matrix
#     yi: label: scalar
#     lam: scalar, the regularization parameter
# Return:
#     obj: scalar, the objective Q_i
#     g: d-by-1 matrix, gradient of Q_i

def stochastic_objective_gradient(w, xi, yi, lam):

  yixi = yi * xi
  yixiw = float(np.dot(yixi,w))

  Obj1 = np.log(1 + np.exp(-yixiw))
  Obj2 = (lam / 2) * np.sum(w * w)

  obj = Obj1 + Obj2

  G1 = -yixi.T/(1 + np.exp(yixiw))
  G2 = lam * w

  g = G1 + G2
  return obj, g

  pass
```

Hints:

1. In every epoch, randomly permute the $n$ samples.
2. Each epoch has $n$ iterations. In every iteration, use 1 sample, and compute the gradient and objective using the `stochastic_objective_gradient` function. In the next iteration, use the next sample, and so on.

In [12]:
```
# SGD for solving logistic regression
# You will need to do iterative process (loops) to obtain optimal weights in t

# Inputs:
#     x: data: n-by-d matrix
#     y: label: n-by-1 matrix
#     lam: scalar, the regularization parameter
#     learning_rate: scalar
#     w: weights: d-by-1 matrix, initialization of w
#     max_epoch: integer, the maximal epochs
# Return:
#
#     w: weights: d-by-1 matrix, the solution
#     objvals: a record of each epoch's objective value
#     Record one objective value per epoch (not per iteration)

def sgd(x, y, lam, learning_rate, w, max_epoch=100):

  n = x.shape[0]
  objvals = np.zeros(max_epoch)

  for i in range(0, max_epoch):

    # shuffling
    rndm_Indices = np.random.permutation(n)
    x_rndm, y_rndm = x[rndm_Indices, :], y[rndm_Indices, :]

    current_Obj_Value = 0
    for j in range(0,n):
      xi, yi = x_rndm[j, :].reshape(1,d), float(y_rndm[j, :])
      curr_Obj, g = stochastic_objective_gradient(w, xi, yi, lam)
      w -= learning_rate * g # update weights
      current_Obj_Value += curr_Obj

    learning_rate *= 0.95
    current_Obj_Value /= n
    objvals[i] = current_Obj_Value
    print(f'Iteration number: {i+1} Current Objective Value: {objvals[i]}')

  return w, objvals

  pass
```

Use sgd function to obtain your optimal weights and a list of objective values over each epoch.

In [13]:
```
# Train logistic regression
# You should get the optimal weights and a list of objective values by using g
lam = 0
learning_rate = 1
w = np.zeros((d,1))
w_SGD, obj_value_sgd = sgd(x_train, y_train, lam, learning_rate, w)
```

```
Iteration number: 1 Current Objective Value: 0.45904857171838404
Iteration number: 2 Current Objective Value: 0.22152348823178272
Iteration number: 3 Current Objective Value: 0.2583501309476069
Iteration number: 4 Current Objective Value: 0.25868129260102374
Iteration number: 5 Current Objective Value: 0.22818288056537978
Iteration number: 6 Current Objective Value: 0.1384057283463276
Iteration number: 7 Current Objective Value: 0.11328067339437724
Iteration number: 8 Current Objective Value: 0.19218315602494768
Iteration number: 9 Current Objective Value: 0.06523637932927986
Iteration number: 10 Current Objective Value: 0.1638489682929377
Iteration number: 11 Current Objective Value: 0.12081470753779183
Iteration number: 12 Current Objective Value: 0.1083013128937812
Iteration number: 13 Current Objective Value: 0.05473336311197603
Iteration number: 14 Current Objective Value: 0.04797114133448048
Iteration number: 15 Current Objective Value: 0.0523828423336764
Iteration number: 16 Current Objective Value: 0.052080012670332666
Iteration number: 17 Current Objective Value: 0.04116372509178781
Iteration number: 18 Current Objective Value: 0.039231037516534714
Iteration number: 19 Current Objective Value: 0.08260145803486879
Iteration number: 20 Current Objective Value: 0.034701376521373374
Iteration number: 21 Current Objective Value: 0.02650272803548619
Iteration number: 22 Current Objective Value: 0.030056112915736847
Iteration number: 23 Current Objective Value: 0.02708335440458205
Iteration number: 24 Current Objective Value: 0.039092523206772686
Iteration number: 25 Current Objective Value: 0.03494370694053452
Iteration number: 26 Current Objective Value: 0.03120430455294283
Iteration number: 27 Current Objective Value: 0.041988828062890574
Iteration number: 28 Current Objective Value: 0.02217953255958025
Iteration number: 29 Current Objective Value: 0.021374011082529364
Iteration number: 30 Current Objective Value: 0.020396158105255306
Iteration number: 31 Current Objective Value: 0.025379294505125903
Iteration number: 32 Current Objective Value: 0.019716862778981388
Iteration number: 33 Current Objective Value: 0.01867862838453151
Iteration number: 34 Current Objective Value: 0.017460679005149863
Iteration number: 35 Current Objective Value: 0.019934122207267136
Iteration number: 36 Current Objective Value: 0.018475890081850865
Iteration number: 37 Current Objective Value: 0.017186965372519798
Iteration number: 38 Current Objective Value: 0.016342459398972137
Iteration number: 39 Current Objective Value: 0.017126202609189164
Iteration number: 40 Current Objective Value: 0.015794253472026086
Iteration number: 41 Current Objective Value: 0.01585793212611267
Iteration number: 42 Current Objective Value: 0.015390861865699572
Iteration number: 43 Current Objective Value: 0.015212059542145913
Iteration number: 44 Current Objective Value: 0.014876243569307777
Iteration number: 45 Current Objective Value: 0.014750892241743354
Iteration number: 46 Current Objective Value: 0.014712664280344338
Iteration number: 47 Current Objective Value: 0.014322380988719215
Iteration number: 48 Current Objective Value: 0.013952667358322799
Iteration number: 49 Current Objective Value: 0.013896413695932391
Iteration number: 50 Current Objective Value: 0.013832880290185817
Iteration number: 51 Current Objective Value: 0.013668088429231751
Iteration number: 52 Current Objective Value: 0.01341287772986861
Iteration number: 53 Current Objective Value: 0.013226052884195533
Iteration number: 54 Current Objective Value: 0.0132654197685369
Iteration number: 55 Current Objective Value: 0.013167605758723519
Iteration number: 56 Current Objective Value: 0.012878539471128934
Iteration number: 57 Current Objective Value: 0.012908296082302316
```

```
Iteration number: 58 Current Objective Value: 0.012812985999654867
Iteration number: 59 Current Objective Value: 0.012715584948152168
Iteration number: 60 Current Objective Value: 0.012590243512441236
Iteration number: 61 Current Objective Value: 0.012511805387232832
Iteration number: 62 Current Objective Value: 0.012508211483513807
Iteration number: 63 Current Objective Value: 0.012414140776022983
Iteration number: 64 Current Objective Value: 0.012320606921097033
Iteration number: 65 Current Objective Value: 0.012274811261156935
Iteration number: 66 Current Objective Value: 0.012199520803819435
Iteration number: 67 Current Objective Value: 0.012139552164600761
Iteration number: 68 Current Objective Value: 0.012105028181584493
Iteration number: 69 Current Objective Value: 0.012070020915759273
Iteration number: 70 Current Objective Value: 0.012005675802297716
Iteration number: 71 Current Objective Value: 0.011985575255835323
Iteration number: 72 Current Objective Value: 0.011926066316003448
Iteration number: 73 Current Objective Value: 0.011886420171971314
Iteration number: 74 Current Objective Value: 0.011875696008044076
Iteration number: 75 Current Objective Value: 0.01184042815829798
Iteration number: 76 Current Objective Value: 0.011796834618966563
Iteration number: 77 Current Objective Value: 0.01176211512517781
Iteration number: 78 Current Objective Value: 0.01174170798506595
Iteration number: 79 Current Objective Value: 0.011713742268807639
Iteration number: 80 Current Objective Value: 0.011689318262077843
Iteration number: 81 Current Objective Value: 0.011663434718797395
Iteration number: 82 Current Objective Value: 0.011642754645378388
Iteration number: 83 Current Objective Value: 0.011614690750274173
Iteration number: 84 Current Objective Value: 0.011592419273889994
Iteration number: 85 Current Objective Value: 0.011576258513866686
Iteration number: 86 Current Objective Value: 0.011555754766797103
Iteration number: 87 Current Objective Value: 0.01154066638299633
Iteration number: 88 Current Objective Value: 0.011525763353020232
Iteration number: 89 Current Objective Value: 0.011508935565311317
Iteration number: 90 Current Objective Value: 0.011492577158205666
Iteration number: 91 Current Objective Value: 0.011480289265560335
Iteration number: 92 Current Objective Value: 0.011467443100622058
Iteration number: 93 Current Objective Value: 0.01145212974390092
Iteration number: 94 Current Objective Value: 0.01144224645353837
Iteration number: 95 Current Objective Value: 0.011428478961409827
Iteration number: 96 Current Objective Value: 0.011419028582724868
Iteration number: 97 Current Objective Value: 0.011408250909638658
Iteration number: 98 Current Objective Value: 0.01139911454985806
Iteration number: 99 Current Objective Value: 0.011389138865918465
Iteration number: 100 Current Objective Value: 0.011381747114136696
```

In [14]:
```
# Train regularized logistic regression
# You should get the optimal weights and a list of objective values by using g
lam = 0.000001
learning_rate = 1
w = np.zeros((x_train.shape[1],1))
w_SGD_REG, obj_value_sgd_reg = sgd(x_train, y_train, lam, learning_rate, w)
```

```
Iteration number: 1 Current Objective Value: 0.3988947379627104
Iteration number: 2 Current Objective Value: 0.17139648949916883
Iteration number: 3 Current Objective Value: 0.22091449122942583
Iteration number: 4 Current Objective Value: 0.24520748005367596
Iteration number: 5 Current Objective Value: 0.14321260998599797
Iteration number: 6 Current Objective Value: 0.11552545824764535
Iteration number: 7 Current Objective Value: 0.09919456929735225
Iteration number: 8 Current Objective Value: 0.06455883366805548
Iteration number: 9 Current Objective Value: 0.13911246777802588
Iteration number: 10 Current Objective Value: 0.23772784772152358
Iteration number: 11 Current Objective Value: 0.16016925979942515
Iteration number: 12 Current Objective Value: 0.09831801795262377
Iteration number: 13 Current Objective Value: 0.11505679197202075
Iteration number: 14 Current Objective Value: 0.06502022434031544
Iteration number: 15 Current Objective Value: 0.08522962366872402
Iteration number: 16 Current Objective Value: 0.07599231480009719
Iteration number: 17 Current Objective Value: 0.06457753837645581
Iteration number: 18 Current Objective Value: 0.04217370079848827
Iteration number: 19 Current Objective Value: 0.03875265857705929
Iteration number: 20 Current Objective Value: 0.042053373898980416
Iteration number: 21 Current Objective Value: 0.030410147526828617
Iteration number: 22 Current Objective Value: 0.027802126282763268
Iteration number: 23 Current Objective Value: 0.028001465318627884
Iteration number: 24 Current Objective Value: 0.046694212914109114
Iteration number: 25 Current Objective Value: 0.02654122202486378
Iteration number: 26 Current Objective Value: 0.02804252295249528
Iteration number: 27 Current Objective Value: 0.02303663668977406
Iteration number: 28 Current Objective Value: 0.022232628792541176
Iteration number: 29 Current Objective Value: 0.02095113242676496
Iteration number: 30 Current Objective Value: 0.019693473257971196
Iteration number: 31 Current Objective Value: 0.023003403734821473
Iteration number: 32 Current Objective Value: 0.02023184431392606
Iteration number: 33 Current Objective Value: 0.018147975169491192
Iteration number: 34 Current Objective Value: 0.01766951784098862
Iteration number: 35 Current Objective Value: 0.01703910037238065
Iteration number: 36 Current Objective Value: 0.01606156248007289
Iteration number: 37 Current Objective Value: 0.016700879942134483
Iteration number: 38 Current Objective Value: 0.016688586241437178
Iteration number: 39 Current Objective Value: 0.015161754749897527
Iteration number: 40 Current Objective Value: 0.01471531978467571
Iteration number: 41 Current Objective Value: 0.01453958633527937
Iteration number: 42 Current Objective Value: 0.014631459214934268
Iteration number: 43 Current Objective Value: 0.014142560844922002
Iteration number: 44 Current Objective Value: 0.01356312463945726
Iteration number: 45 Current Objective Value: 0.014009280794852955
Iteration number: 46 Current Objective Value: 0.01349606263307861
Iteration number: 47 Current Objective Value: 0.01326864210276023
Iteration number: 48 Current Objective Value: 0.01319478769136293
Iteration number: 49 Current Objective Value: 0.013128906786289057
Iteration number: 50 Current Objective Value: 0.012808859477215575
Iteration number: 51 Current Objective Value: 0.012815832520990163
Iteration number: 52 Current Objective Value: 0.012570286628619623
Iteration number: 53 Current Objective Value: 0.012511398906483413
Iteration number: 54 Current Objective Value: 0.012361585938316171
Iteration number: 55 Current Objective Value: 0.01228511958404225
Iteration number: 56 Current Objective Value: 0.012175431460153095
Iteration number: 57 Current Objective Value: 0.012099760453981364
```

```
Iteration number: 58 Current Objective Value: 0.011972645215117 09
Iteration number: 59 Current Objective Value: 0.011898068544065183
Iteration number: 60 Current Objective Value: 0.011790823754791008
Iteration number: 61 Current Objective Value: 0.011750745800510536
Iteration number: 62 Current Objective Value: 0.011610847654502116
Iteration number: 63 Current Objective Value: 0.011674099054992476
Iteration number: 64 Current Objective Value: 0.011564680698560292
Iteration number: 65 Current Objective Value: 0.011490504586086372
Iteration number: 66 Current Objective Value: 0.011437104841680208
Iteration number: 67 Current Objective Value: 0.01144070508924828
Iteration number: 68 Current Objective Value: 0.011365548902772603
Iteration number: 69 Current Objective Value: 0.011308457759075853
Iteration number: 70 Current Objective Value: 0.01128085027293187
Iteration number: 71 Current Objective Value: 0.011229213445014916
Iteration number: 72 Current Objective Value: 0.011209553137227308
Iteration number: 73 Current Objective Value: 0.011137423914644465
Iteration number: 74 Current Objective Value: 0.011142897169174285
Iteration number: 75 Current Objective Value: 0.011092983896794884
Iteration number: 76 Current Objective Value: 0.011079998611836396
Iteration number: 77 Current Objective Value: 0.011043244635839338
Iteration number: 78 Current Objective Value: 0.01102188645354807
Iteration number: 79 Current Objective Value: 0.010997065744759185
Iteration number: 80 Current Objective Value: 0.010968986473604869
Iteration number: 81 Current Objective Value: 0.010952717756924458
Iteration number: 82 Current Objective Value: 0.01092682562046908
Iteration number: 83 Current Objective Value: 0.010910703669697997
Iteration number: 84 Current Objective Value: 0.010893655116196985
Iteration number: 85 Current Objective Value: 0.010873117371467365
Iteration number: 86 Current Objective Value: 0.010858841739022245
Iteration number: 87 Current Objective Value: 0.01083974834130454
Iteration number: 88 Current Objective Value: 0.010825007698956857
Iteration number: 89 Current Objective Value: 0.010810432729700874
Iteration number: 90 Current Objective Value: 0.010799408752011914
Iteration number: 91 Current Objective Value: 0.010783424697876996
Iteration number: 92 Current Objective Value: 0.010773162434044732
Iteration number: 93 Current Objective Value: 0.01076082912445525
Iteration number: 94 Current Objective Value: 0.010750558460944887
Iteration number: 95 Current Objective Value: 0.010742020330835749
Iteration number: 96 Current Objective Value: 0.010732443860823705
Iteration number: 97 Current Objective Value: 0.010722350345546734
Iteration number: 98 Current Objective Value: 0.01071478109552214
Iteration number: 99 Current Objective Value: 0.010706469111421656
Iteration number: 100 Current Objective Value: 0.010698046751838207
```

## 3.3 Mini-Batch Gradient Descent (MBGD)

Define $Q_I(w) = \frac{1}{b}\sum_{i \in I} \log\left(1 + \exp\left(-y_i x_i^T w\right)\right) + \frac{\lambda}{2}\|w\|_2^2$, where $I$ is a set containing $b$ indices randomly drawn from $\{1, \cdots, n\}$ without replacement.

The stochastic gradient at $w$ is $g_I = \frac{\partial Q_I}{\partial w} = \frac{1}{b}\sum_{i \in I} \frac{-y_i x_i}{1 + \exp\ (y_i x_i^T w)} + \lambda w.$

In [15]:
```
# Calculate the objective Q_I and the gradient of Q_I
# Inputs:
#     w: weights: d-by-b matrix
#     xi: data: b-by-d matrix
#     yi: label: scalar
#     lam: scalar, the regularization parameter
# Return:
#     obj: scalar, the objective Q_i
#     g: d-by-1 matrix, gradient of Q_i

def mb_objective_gradient(w, xi, yi, lam):

  yixi = np.multiply(yi, xi)
  yixiw = np.dot(yixi, w)

  Obj1 = np.mean(np.log(1 + np.exp(-yixiw)))
  Obj2 = (lam/2) * np.sum(w * w)

  obj = Obj1 + Obj2

  G1 = np.mean(np.divide(-yixi, 1 + np.exp(yixiw)), axis = 0).reshape(d,1)
  G2 = lam * w

  g = G1 + G2

  return obj, g

  pass
```

Hints:

1. In every epoch, randomly permute the $n$ samples (just like SGD).
2. Each epoch has $\frac{n}{b}$ iterations. In every iteration, use $b$ samples, and compute the gradient and objective using the `mb_objective_gradient` function. In the next iteration, use the next $b$ samples, and so on.

In [16]:
```
# MBGD for solving logistic regression
# You will need to do iterative process (loops) to obtain optimal weights in t

# Inputs:
#     x: data: n-by-d matrix
#     y: label: n-by-1 matrix
#     lam: scalar, the regularization parameter
#     learning_rate: scalar
#     w: weights: d-by-1 matrix, initialization of w
#     max_epoch: integer, the maximal epochs
# Return:
#     w: weights: d-by-1 matrix, the solution
#     objvals: a record of each epoch's objective value
#     Record one objective value per epoch (not per iteration)

def mbgd(x, y, lam, learning_rate, w, max_epoch=100):

  objvals = np.zeros(max_epoch)
  batch_Size = 10
  n = x.shape[0]

  for i in range(max_epoch) :

    rndm_Indices = np.random.permutation(n)
    x_rndm, y_rndm = x[rndm_Indices, : ], y [rndm_Indices, : ]

    current_Obj_Value = 0
    for j in range(0, n, batch_Size):
      xi, yi = x_rndm[j : j + batch_Size, :], y_rndm[j : j + batch_Size, :]
      curr_Obj, g = mb_objective_gradient(w, xi, yi, lam)
      current_Obj_Value += curr_Obj
      w -= learning_rate * g

    learning_rate *= 0.95
    current_Obj_Value /= (n/batch_Size)
    objvals[i] = current_Obj_Value
    print(f'Iteration number: {i+1} Current Objective Value: {objvals[i]}')


  return w, objvals
  pass
```

Use mbgd function to obtain your optimal weights and a list of objective values over each epoch.

In [17]:
```
# Train logistic regression
# You should get the optimal weights and a list of objective values by using g
lam = 0
learning_rate = 1
w = np.zeros((x_train.shape[1],1))
w_MBGD, obj_value_mbgd = mbgd(x_train, y_train, lam, learning_rate, w)
```

```
Iteration number: 1 Current Objective Value: 0.1138627882382831
Iteration number: 2 Current Objective Value: 0.06282958048524258
Iteration number: 3 Current Objective Value: 0.05435094874163979
Iteration number: 4 Current Objective Value: 0.051298606339948954
Iteration number: 5 Current Objective Value: 0.04830962539636071
Iteration number: 6 Current Objective Value: 0.04597496033118972
Iteration number: 7 Current Objective Value: 0.04452332023684668
Iteration number: 8 Current Objective Value: 0.04346461197927203
Iteration number: 9 Current Objective Value: 0.040554887051474336
Iteration number: 10 Current Objective Value: 0.04127323552804113
Iteration number: 11 Current Objective Value: 0.039370123747865034
Iteration number: 12 Current Objective Value: 0.03820265519211409
Iteration number: 13 Current Objective Value: 0.03739064516063011
Iteration number: 14 Current Objective Value: 0.038098739887347195
Iteration number: 15 Current Objective Value: 0.0367242978357271
Iteration number: 16 Current Objective Value: 0.03629405903982211
Iteration number: 17 Current Objective Value: 0.035893124661632594
Iteration number: 18 Current Objective Value: 0.03617225023926388
Iteration number: 19 Current Objective Value: 0.03517291844736462
Iteration number: 20 Current Objective Value: 0.03465516751318265
Iteration number: 21 Current Objective Value: 0.0345351605163535
Iteration number: 22 Current Objective Value: 0.03408906489303702
Iteration number: 23 Current Objective Value: 0.03489394442128787
Iteration number: 24 Current Objective Value: 0.03387490101493365
Iteration number: 25 Current Objective Value: 0.03377774164704562
Iteration number: 26 Current Objective Value: 0.03345876710779413
Iteration number: 27 Current Objective Value: 0.03307252582603821
Iteration number: 28 Current Objective Value: 0.03283169324032151
Iteration number: 29 Current Objective Value: 0.03292518764823498
Iteration number: 30 Current Objective Value: 0.03265682571711327
Iteration number: 31 Current Objective Value: 0.03241151527468455
Iteration number: 32 Current Objective Value: 0.03250582481170097
Iteration number: 33 Current Objective Value: 0.03218106791165075
Iteration number: 34 Current Objective Value: 0.032200947985677394
Iteration number: 35 Current Objective Value: 0.03214952339640993
Iteration number: 36 Current Objective Value: 0.032193198855658973
Iteration number: 37 Current Objective Value: 0.03212238638662302
Iteration number: 38 Current Objective Value: 0.031637538776829614
Iteration number: 39 Current Objective Value: 0.031676448898615046
Iteration number: 40 Current Objective Value: 0.031567246735790734
Iteration number: 41 Current Objective Value: 0.03138825863768707
Iteration number: 42 Current Objective Value: 0.031936958591240766
Iteration number: 43 Current Objective Value: 0.03177198511624216
Iteration number: 44 Current Objective Value: 0.031153472651312138
Iteration number: 45 Current Objective Value: 0.0310977209822527
Iteration number: 46 Current Objective Value: 0.031212508658404463
Iteration number: 47 Current Objective Value: 0.031182503179323143
Iteration number: 48 Current Objective Value: 0.030899527456243487
Iteration number: 49 Current Objective Value: 0.030804691803135102
Iteration number: 50 Current Objective Value: 0.031233251334662544
Iteration number: 51 Current Objective Value: 0.03073541731732659
Iteration number: 52 Current Objective Value: 0.030691322290362432
Iteration number: 53 Current Objective Value: 0.030673165817557878
Iteration number: 54 Current Objective Value: 0.031155951567057482
Iteration number: 55 Current Objective Value: 0.030600896334475478
Iteration number: 56 Current Objective Value: 0.03061098981510988
Iteration number: 57 Current Objective Value: 0.030514384361790155
```

```
Iteration number: 58 Current Objective Value: 0.03075645024640464
Iteration number: 59 Current Objective Value: 0.03043690184325178
Iteration number: 60 Current Objective Value: 0.030394269177487176
Iteration number: 61 Current Objective Value: 0.03160082209753617
Iteration number: 62 Current Objective Value: 0.030341548895448697
Iteration number: 63 Current Objective Value: 0.030337816499462687
Iteration number: 64 Current Objective Value: 0.030621681180728305
Iteration number: 65 Current Objective Value: 0.030471744774963574
Iteration number: 66 Current Objective Value: 0.03106970751918138
Iteration number: 67 Current Objective Value: 0.030477519542323854
Iteration number: 68 Current Objective Value: 0.030234582904842038
Iteration number: 69 Current Objective Value: 0.030640660881183734
Iteration number: 70 Current Objective Value: 0.03268482256863131
Iteration number: 71 Current Objective Value: 0.030161626737904863
Iteration number: 72 Current Objective Value: 0.030227359793570757
Iteration number: 73 Current Objective Value: 0.030148250715913354
Iteration number: 74 Current Objective Value: 0.032557837527664
Iteration number: 75 Current Objective Value: 0.030589472150604678
Iteration number: 76 Current Objective Value: 0.030096495659505066
Iteration number: 77 Current Objective Value: 0.03079790648367475
Iteration number: 78 Current Objective Value: 0.030072607539369408
Iteration number: 79 Current Objective Value: 0.030457502601079132
Iteration number: 80 Current Objective Value: 0.030540314415347002
Iteration number: 81 Current Objective Value: 0.0302754775082596
Iteration number: 82 Current Objective Value: 0.030032745519567167
Iteration number: 83 Current Objective Value: 0.030030654462559977
Iteration number: 84 Current Objective Value: 0.030007667046777518
Iteration number: 85 Current Objective Value: 0.03009147244549239
Iteration number: 86 Current Objective Value: 0.030040998691829746
Iteration number: 87 Current Objective Value: 0.030246298262996855
Iteration number: 88 Current Objective Value: 0.035161879934573675
Iteration number: 89 Current Objective Value: 0.030344402815520418
Iteration number: 90 Current Objective Value: 0.030274549420702422
Iteration number: 91 Current Objective Value: 0.029998052978395227
Iteration number: 92 Current Objective Value: 0.030066395707889892
Iteration number: 93 Current Objective Value: 0.029965745834625246
Iteration number: 94 Current Objective Value: 0.029973778860998838
Iteration number: 95 Current Objective Value: 0.029943520228605792
Iteration number: 96 Current Objective Value: 0.030040993996528074
Iteration number: 97 Current Objective Value: 0.029936458729780617
Iteration number: 98 Current Objective Value: 0.029947464539859102
Iteration number: 99 Current Objective Value: 0.030674442189695866
Iteration number: 100 Current Objective Value: 0.029921237090743995
```

In [18]:
```
# Train regularized logistic regression
# You should get the optimal weights and a list of objective values by using g
lam = 0.000001
learning_rate = 1
w = np.zeros((x_train.shape[1],1))
w_MBGD_REG, obj_value_mbgd_reg = mbgd(x_train, y_train, lam, learning_rate, w)
```

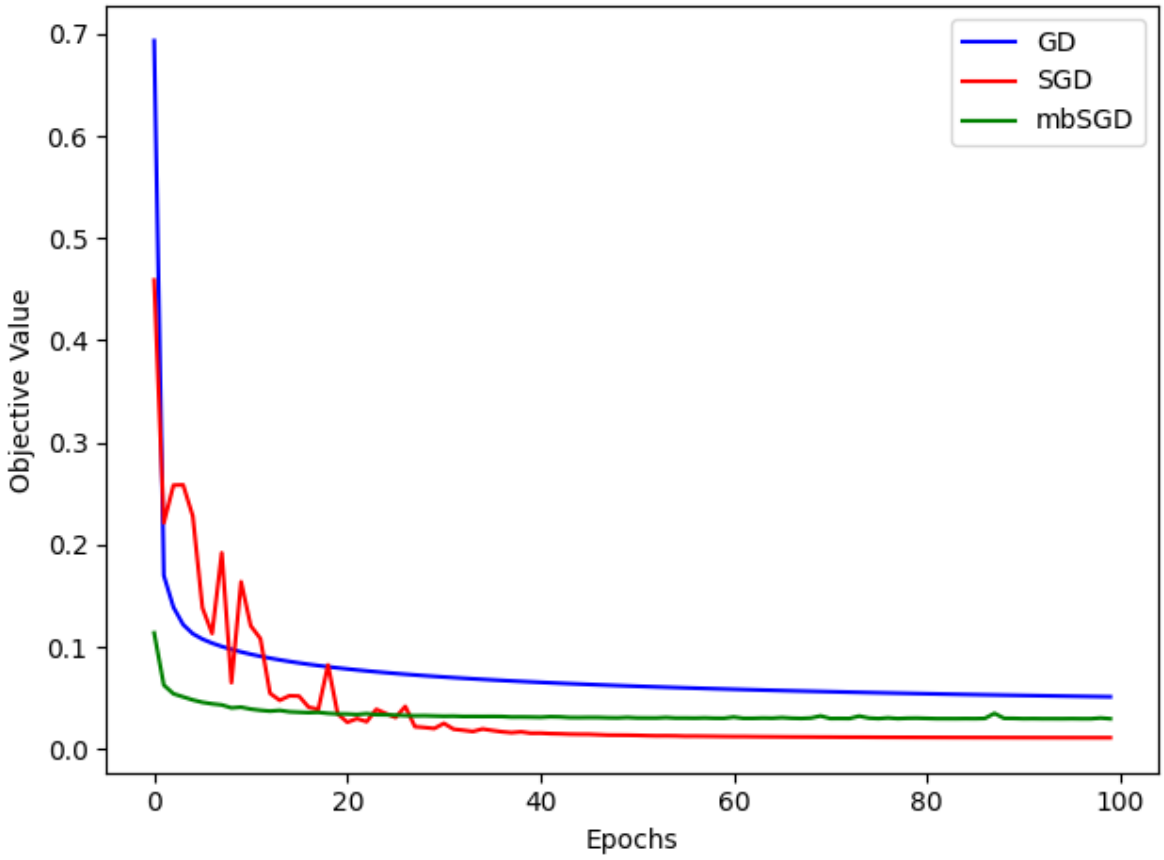
```
Iteration number: 1 Current Objective Value: 0.11104688621520847
Iteration number: 2 Current Objective Value: 0.06671746889823636
Iteration number: 3 Current Objective Value: 0.05750323736178438
Iteration number: 4 Current Objective Value: 0.05545493119875436
Iteration number: 5 Current Objective Value: 0.04855914494414417
Iteration number: 6 Current Objective Value: 0.04754946388400939
Iteration number: 7 Current Objective Value: 0.04329673884466853
Iteration number: 8 Current Objective Value: 0.041855221873335756
Iteration number: 9 Current Objective Value: 0.041096448522545695
Iteration number: 10 Current Objective Value: 0.03952401533576728
Iteration number: 11 Current Objective Value: 0.03948975046148922
Iteration number: 12 Current Objective Value: 0.03847858404897913
Iteration number: 13 Current Objective Value: 0.03850242904467933
Iteration number: 14 Current Objective Value: 0.03823970524268212
Iteration number: 15 Current Objective Value: 0.03767071164240979
Iteration number: 16 Current Objective Value: 0.03592098994715343
Iteration number: 17 Current Objective Value: 0.035681657110516955
Iteration number: 18 Current Objective Value: 0.035649084170529484
Iteration number: 19 Current Objective Value: 0.035258981791037695
Iteration number: 20 Current Objective Value: 0.03478311574543653
Iteration number: 21 Current Objective Value: 0.03626420505305452
Iteration number: 22 Current Objective Value: 0.03494314512668418
Iteration number: 23 Current Objective Value: 0.0350586642623087
Iteration number: 24 Current Objective Value: 0.03381032787796235
Iteration number: 25 Current Objective Value: 0.03342605223639978
Iteration number: 26 Current Objective Value: 0.03359627991903764
Iteration number: 27 Current Objective Value: 0.03306384567122108
Iteration number: 28 Current Objective Value: 0.03313358746142331
Iteration number: 29 Current Objective Value: 0.03301992422351711
Iteration number: 30 Current Objective Value: 0.03269273491185629
Iteration number: 31 Current Objective Value: 0.032686732062850256
Iteration number: 32 Current Objective Value: 0.03342975859087475
Iteration number: 33 Current Objective Value: 0.032256064195469725
Iteration number: 34 Current Objective Value: 0.03215286046182417
Iteration number: 35 Current Objective Value: 0.032169997732783215
Iteration number: 36 Current Objective Value: 0.03186129820600132
Iteration number: 37 Current Objective Value: 0.03284186397543271
Iteration number: 38 Current Objective Value: 0.031847831099125486
Iteration number: 39 Current Objective Value: 0.03232136717634626
Iteration number: 40 Current Objective Value: 0.0318992952618288
Iteration number: 41 Current Objective Value: 0.03150549136651267
Iteration number: 42 Current Objective Value: 0.03144467888253223
Iteration number: 43 Current Objective Value: 0.03159373255867552
Iteration number: 44 Current Objective Value: 0.031185491641309633
Iteration number: 45 Current Objective Value: 0.031135495525480204
Iteration number: 46 Current Objective Value: 0.03199392618573883
Iteration number: 47 Current Objective Value: 0.031443410340987965
Iteration number: 48 Current Objective Value: 0.030964811654714142
Iteration number: 49 Current Objective Value: 0.030924652310511506
Iteration number: 50 Current Objective Value: 0.03132265489911997
Iteration number: 51 Current Objective Value: 0.03081708011360181
Iteration number: 52 Current Objective Value: 0.030767407066504826
Iteration number: 53 Current Objective Value: 0.030713943151158343
Iteration number: 54 Current Objective Value: 0.03098838997068119
Iteration number: 55 Current Objective Value: 0.03076728527994163
Iteration number: 56 Current Objective Value: 0.03060977069835814
Iteration number: 57 Current Objective Value: 0.030683746735425795
```

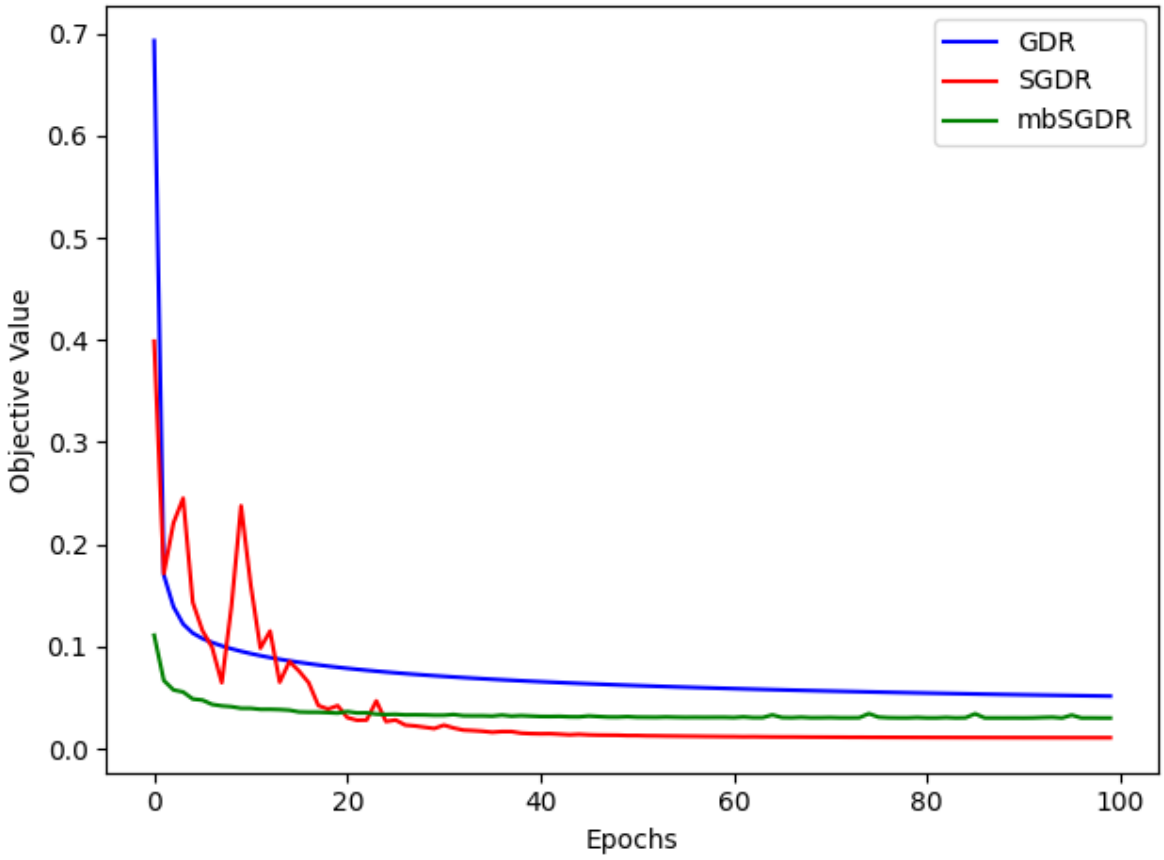
```
Iteration number: 58 Current Objective Value: 0.030624150111174673
Iteration number: 59 Current Objective Value: 0.03067604726911029
Iteration number: 60 Current Objective Value: 0.030698949921167604
Iteration number: 61 Current Objective Value: 0.03044913829306839
Iteration number: 62 Current Objective Value: 0.031095032052091295
Iteration number: 63 Current Objective Value: 0.030413634586365357
Iteration number: 64 Current Objective Value: 0.03039773881813939
Iteration number: 65 Current Objective Value: 0.03301554085302835
Iteration number: 66 Current Objective Value: 0.030504611397765027
Iteration number: 67 Current Objective Value: 0.030315938934148474
Iteration number: 68 Current Objective Value: 0.03076720625250832
Iteration number: 69 Current Objective Value: 0.030274332635768668
Iteration number: 70 Current Objective Value: 0.03031061614213802
Iteration number: 71 Current Objective Value: 0.030478269447869643
Iteration number: 72 Current Objective Value: 0.030219436685651063
Iteration number: 73 Current Objective Value: 0.030204890981413192
Iteration number: 74 Current Objective Value: 0.030275760885183776
Iteration number: 75 Current Objective Value: 0.03427218790003496
Iteration number: 76 Current Objective Value: 0.030832722933968295
Iteration number: 77 Current Objective Value: 0.030307949931564642
Iteration number: 78 Current Objective Value: 0.03016692881496787
Iteration number: 79 Current Objective Value: 0.0301556903990799
Iteration number: 80 Current Objective Value: 0.030517492726929052
Iteration number: 81 Current Objective Value: 0.030115694061608436
Iteration number: 82 Current Objective Value: 0.030111682759741014
Iteration number: 83 Current Objective Value: 0.030525307680675292
Iteration number: 84 Current Objective Value: 0.03008910523751723
Iteration number: 85 Current Objective Value: 0.030256544175669555
Iteration number: 86 Current Objective Value: 0.03409333118497612
Iteration number: 87 Current Objective Value: 0.03007803000483531
Iteration number: 88 Current Objective Value: 0.030064853084661985
Iteration number: 89 Current Objective Value: 0.030054830647984993
Iteration number: 90 Current Objective Value: 0.030104125172404524
Iteration number: 91 Current Objective Value: 0.030052583543107903
Iteration number: 92 Current Objective Value: 0.030169242182657017
Iteration number: 93 Current Objective Value: 0.030520251474394345
Iteration number: 94 Current Objective Value: 0.030761166328354374
Iteration number: 95 Current Objective Value: 0.030023402476312744
Iteration number: 96 Current Objective Value: 0.0328297259910142
Iteration number: 97 Current Objective Value: 0.030048867189599572
Iteration number: 98 Current Objective Value: 0.030077704157709672
Iteration number: 99 Current Objective Value: 0.03000503278644595
Iteration number: 100 Current Objective Value: 0.030000917445201208
```

# 4. Compare GD, SGD, MBGD

### Plot objective function values against epochs.

In [19]:
```
import matplotlib.pyplot as plt
%matplotlib inline

max_epoch = 100

line0, = plt.plot(range(0,max_epoch), obj_value_gd, '-b')
line1, = plt.plot(range(0,max_epoch), obj_value_sgd, '-r')
line2, = plt.plot(range(0,max_epoch), obj_value_mbgd, '-g')

plt.ylabel('Objective Value', fontsize=10)
plt.xlabel('Epochs', fontsize=10)
plt.yticks(fontsize=10)
plt.xticks(fontsize=10)
plt.legend([line0, line1, line2], ['GD', 'SGD', 'mbSGD'], fontsize=10)
plt.tight_layout()
plt.show()

line0, = plt.plot(range(0,max_epoch), obj_value_gd_reg, '-b')
line1, = plt.plot(range(0,max_epoch), obj_value_sgd_reg, '-r')
line2, = plt.plot(range(0,max_epoch), obj_value_mbgd_reg, '-g')

plt.ylabel('Objective Value', fontsize=10)
plt.xlabel('Epochs', fontsize=10)
plt.yticks(fontsize=10)
plt.xticks(fontsize=10)
plt.legend([line0, line1, line2], ['GDR', 'SGDR', 'mbSGDR'], fontsize=10)
plt.tight_layout()
plt.show()
```




# 5. Prediction

## Compare the training and testing accuracy for logistic regression and regularized logistic regression.

In [20]:
```
# Predict class label
# Inputs:
#     w: weights: d-by-1 matrix
#     X: data: m-by-d matrix
# Return:
#     f: m-by-1 matrix, the predictions
def predict(w, X):
  f = np.sign(np.dot(X,w))
  return np.asarray(f)
  pass
```

In [21]:
```
# evaluate training error of logistic regression and regularized version

# gradient descent
pred_gd = predict(w_GD, x_train)
Error = numpy.abs(pred_gd - y_train) / 2
error_pred_gd = numpy.mean(Error)
print('Error in Training dataset for Gradient Descent is ' + str(error_pred_gd

# stochastic gradient descent
pred_sgd = predict(w_SGD, x_train)
Error = numpy.abs(pred_sgd - y_train) / 2
error_pred_sgd = numpy.mean(Error)
print('Error in Training dataset for Stochastic Gradient Descent is ' + str(er

# mini batch gradient descent
pred_mbgd = predict(w_MBGD, x_train)
Error = numpy.abs(pred_mbgd - y_train) / 2
error_pred_mbgd = numpy.mean(Error)
print('Error in Training dataset for Mini-Batch Gradient Descent is ' + str(er

# gradient descent(regularized)
pred_gd_reg = predict(w_GD_REG, x_train)
Error = numpy.abs(pred_gd_reg - y_train) / 2
error_pred_gd_reg = numpy.mean(Error)
print('Error in Training dataset for Gradient Descent (Regularized) is ' + str

# stochastic gradient descent(regularized)
pred_sgd_reg = predict(w_SGD_REG, x_train)
Error = numpy.abs(pred_sgd_reg - y_train) / 2
error_pred_sgd_reg = numpy.mean(Error)
print('Error in Training dataset for Stochastic Gradient Descent (Regularized)

# mini batch gradient descent(regularized)
pred_mbgd_reg = predict(w_MBGD_REG, x_train)
Error = numpy.abs(pred_mbgd_reg - y_train) / 2
error_pred_mbgd_reg = numpy.mean(Error)
print('Error in Training dataset for Mini-Batch Gradient Descent (Regularized)
```

```
Error in Training dataset for Gradient Descent is 0.013186813186813187
Error in Training dataset for Stochastic Gradient Descent is 0.00219780219780
2198
Error in Training dataset for Mini-Batch Gradient Descent is 0.00879120879120
8791
Error in Training dataset for Gradient Descent (Regularized) is 0.01318681318
6813187
Error in Training dataset for Stochastic Gradient Descent (Regularized) is 0.
002197802197802198
Error in Training dataset for Mini-Batch Gradient Descent (Regularized) is 0.
008791208791208791
```

In [22]:
```
# evaluate testing error of logistic regression and regularized version

# gradient descent
pred_gd = predict(w_GD, x_test)
Error = numpy.abs(pred_gd - y_test) / 2
error_pred_gd = numpy.mean(Error)
print('Error in Testing dataset for Gradient Descent is ' + str(error_pred_gd)

# stochastic gradient descent
pred_sgd = predict(w_SGD, x_test)
Error = numpy.abs(pred_sgd - y_test) / 2
error_pred_sgd = numpy.mean(Error)
print('Error in Testing dataset for Stochastic Gradient Descent is ' + str(err

# mini batch gradient descent
pred_mbgd = predict(w_MBGD, x_test)
Error = numpy.abs(pred_mbgd - y_test) / 2
error_pred_mbgd = numpy.mean(Error)
print('Error in Testing dataset for Mini-Batch Gradient Descent is ' + str(err

# gradient descent(regularized)
pred_gd_reg = predict(w_GD_REG, x_test)
Error = numpy.abs(pred_gd_reg - y_test) / 2
error_pred_gd_reg = numpy.mean(Error)
print('Error in Testing dataset for Gradient Descent (Regularized) is ' + str(

# stochastic gradient descent(regularized)
pred_sgd_reg = predict(w_SGD_REG, x_test)
Error = numpy.abs(pred_sgd_reg - y_test) / 2
error_pred_sgd_reg = numpy.mean(Error)
print('Error in Testing dataset for Stochastic Gradient Descent (Regularized)

# mini batch gradient descent(regularized)
pred_mbgd_reg = predict(w_MBGD_REG, x_test)
Error = numpy.abs(pred_mbgd_reg - y_test) / 2
error_pred_mbgd_reg = numpy.mean(Error)
print('Error in Testing dataset for Mini-Batch Gradient Descent (Regularized)
```

```
Error in Testing dataset for Gradient Descent is 0.02631578947368421
Error in Testing dataset for Stochastic Gradient Descent is 0.026315789473684
21
Error in Testing dataset for Mini-Batch Gradient Descent is 0.026315789473684
21
Error in Testing dataset for Gradient Descent (Regularized) is 0.026315789473
68421
Error in Testing dataset for Stochastic Gradient Descent (Regularized) is 0.0
2631578947368421
Error in Testing dataset for Mini-Batch Gradient Descent (Regularized) is 0.0
2631578947368421
```

# 6. Parameters tuning

## In this section, you may try different combinations of parameters (regularization value, learning rate, etc) to see their effects on the model. (Open ended question)

In [23]:
```
lam = 0.0001
learning_rate = 1
w = np.zeros((d,1))
```

In [24]:
```
w_GD, obj_value_gd = gradient_descent(x_train, y_train, lam, learning_rate, w,
```

```
Iteration number: 1 Current Objective Value: 0.6931471805599453
Iteration number: 2 Current Objective Value: 0.17012213406611762
Iteration number: 3 Current Objective Value: 0.13906400077411965
Iteration number: 4 Current Objective Value: 0.12213880603762639
Iteration number: 5 Current Objective Value: 0.11322276464307639
Iteration number: 6 Current Objective Value: 0.10798362104967402
Iteration number: 7 Current Objective Value: 0.10406004676410457
Iteration number: 8 Current Objective Value: 0.10080035824642906
Iteration number: 9 Current Objective Value: 0.09798030139304911
Iteration number: 10 Current Objective Value: 0.09548993636241487
Iteration number: 11 Current Objective Value: 0.09326195001172602
Iteration number: 12 Current Objective Value: 0.09124966892246412
Iteration number: 13 Current Objective Value: 0.08941839844750421
Iteration number: 14 Current Objective Value: 0.08774126134414528
Iteration number: 15 Current Objective Value: 0.08619687283017277
Iteration number: 16 Current Objective Value: 0.08476789361040797
Iteration number: 17 Current Objective Value: 0.08344005995753992
Iteration number: 18 Current Objective Value: 0.0822014997711691
Iteration number: 19 Current Objective Value: 0.08104223292684103
```

In [25]:
```
w_sGD, obj_value_sgd = sgd(x_train, y_train, lam, learning_rate, w, 1000)
```

```
Iteration number: 1 Current Objective Value: 0.27860301867210124
Iteration number: 2 Current Objective Value: 0.18223946045709594
Iteration number: 3 Current Objective Value: 0.3207869214400415
Iteration number: 4 Current Objective Value: 0.1933384004597344
Iteration number: 5 Current Objective Value: 0.1956433171871641
Iteration number: 6 Current Objective Value: 0.2387774534751289
Iteration number: 7 Current Objective Value: 0.13166200594635455
Iteration number: 8 Current Objective Value: 0.17770691945632586
Iteration number: 9 Current Objective Value: 0.15542927923629632
Iteration number: 10 Current Objective Value: 0.1326725684059027
Iteration number: 11 Current Objective Value: 0.1109836232569448
Iteration number: 12 Current Objective Value: 0.08446655107169321
Iteration number: 13 Current Objective Value: 0.06469878552448614
Iteration number: 14 Current Objective Value: 0.09102293242317264
Iteration number: 15 Current Objective Value: 0.1389410045681275
Iteration number: 16 Current Objective Value: 0.07004082085027302
Iteration number: 17 Current Objective Value: 0.06447478256505688
Iteration number: 18 Current Objective Value: 0.05360858925876394
Iteration number: 19 Current Objective Value: 0.05238369887421473
```

In [26]:
```
w_MBGD, obj_value_mbgd = mbgd(x_train, y_train, lam, learning_rate, w, 1000)
```

```
Iteration number: 1 Current Objective Value: 0.0338027541085031
Iteration number: 2 Current Objective Value: 0.035461820290898505
Iteration number: 3 Current Objective Value: 0.03197576521419303
Iteration number: 4 Current Objective Value: 0.03026167844076463
Iteration number: 5 Current Objective Value: 0.030843833084609144
Iteration number: 6 Current Objective Value: 0.029932744593259075
Iteration number: 7 Current Objective Value: 0.03436642422336133
Iteration number: 8 Current Objective Value: 0.03037217671925373
Iteration number: 9 Current Objective Value: 0.029776537375464568
Iteration number: 10 Current Objective Value: 0.029283357234477368
Iteration number: 11 Current Objective Value: 0.029780426695112144
Iteration number: 12 Current Objective Value: 0.029341703012891244
Iteration number: 13 Current Objective Value: 0.02922532158026563
Iteration number: 14 Current Objective Value: 0.029294088876465844
Iteration number: 15 Current Objective Value: 0.029278097545756415
Iteration number: 16 Current Objective Value: 0.028866051405618227
Iteration number: 17 Current Objective Value: 0.02896043053567271
Iteration number: 18 Current Objective Value: 0.028938718720874657
Iteration number: 19 Current Objective Value: 0.028994198472804104
```

In [27]:
```
w_GD_REG, obj_value_gd_reg = gradient_descent(x_train, y_train, lam, learning
```

```
Iteration number: 1 Current Objective Value: 0.027503306700403997
Iteration number: 2 Current Objective Value: 0.027502412901470038
Iteration number: 3 Current Objective Value: 0.02750152122903432
Iteration number: 4 Current Objective Value: 0.027500631605250996
Iteration number: 5 Current Objective Value: 0.027499743956325005
Iteration number: 6 Current Objective Value: 0.027498858212258833
Iteration number: 7 Current Objective Value: 0.027497974306618878
Iteration number: 8 Current Objective Value: 0.027497092176319626
Iteration number: 9 Current Objective Value: 0.027496211761423858
Iteration number: 10 Current Objective Value: 0.027495333004957684
Iteration number: 11 Current Objective Value: 0.027494455852738804
Iteration number: 12 Current Objective Value: 0.0274935802532171
Iteration number: 13 Current Objective Value: 0.027492706157326168
Iteration number: 14 Current Objective Value: 0.027491833518345228
Iteration number: 15 Current Objective Value: 0.02749096229177017
Iteration number: 16 Current Objective Value: 0.027490092435193264
Iteration number: 17 Current Objective Value: 0.027489223908190648
Iteration number: 18 Current Objective Value: 0.02748835667221705
Iteration number: 19 Current Objective Value: 0.027487490690507106
```

In [28]:
```
w_SGD_REG, obj_value_sgd_reg = sgd(x_train, y_train, lam, learning_rate, w, 10
```

```
Iteration number: 1 Current Objective Value: 0.271209388333179
Iteration number: 2 Current Objective Value: 0.112554947189954
Iteration number: 3 Current Objective Value: 0.23515909148716269
Iteration number: 4 Current Objective Value: 0.2869147609802947
Iteration number: 5 Current Objective Value: 0.1840910202925611
Iteration number: 6 Current Objective Value: 0.14649946365558933
Iteration number: 7 Current Objective Value: 0.11107580718948704
Iteration number: 8 Current Objective Value: 0.06473912320687786
Iteration number: 9 Current Objective Value: 0.29296847287342925
Iteration number: 10 Current Objective Value: 0.1511530031710068
Iteration number: 11 Current Objective Value: 0.11017201685736412
Iteration number: 12 Current Objective Value: 0.12355529529123252
Iteration number: 13 Current Objective Value: 0.11504287669786724
Iteration number: 14 Current Objective Value: 0.12935083888916393
Iteration number: 15 Current Objective Value: 0.09884557344949893
Iteration number: 16 Current Objective Value: 0.0846333304418943
Iteration number: 17 Current Objective Value: 0.0633560938546884
Iteration number: 18 Current Objective Value: 0.05551512183343589
Iteration number: 19 Current Objective Value: 0.04969725630243232
```

In [29]:
```
w_MBGD_REG, obj_value_mbgd_reg = mbgd(x_train, y_train, lam, learning_rate, w,
```

```
Iteration number: 1 Current Objective Value: 0.032644090495445345
Iteration number: 2 Current Objective Value: 0.03223086438972827
Iteration number: 3 Current Objective Value: 0.03170693728459496
Iteration number: 4 Current Objective Value: 0.0309744392571592
Iteration number: 5 Current Objective Value: 0.0307813081772001
Iteration number: 6 Current Objective Value: 0.031304374873774896
Iteration number: 7 Current Objective Value: 0.030540120236407373
Iteration number: 8 Current Objective Value: 0.03019166430415967
Iteration number: 9 Current Objective Value: 0.030193680019769912
Iteration number: 10 Current Objective Value: 0.03030971670470353
Iteration number: 11 Current Objective Value: 0.03005488392984844
Iteration number: 12 Current Objective Value: 0.029898499672243617
Iteration number: 13 Current Objective Value: 0.03015862512841512
Iteration number: 14 Current Objective Value: 0.030161284219468548
Iteration number: 15 Current Objective Value: 0.029887258666410867
Iteration number: 16 Current Objective Value: 0.02979936697447153
Iteration number: 17 Current Objective Value: 0.029679461740395256
Iteration number: 18 Current Objective Value: 0.02977510264679995
Iteration number: 19 Current Objective Value: 0.029468537166253424
```