21221030_Lab_4

November 21, 2024

1 EEE385L: Machine Learning Laboratory

2 Lab 4: Implementation of Logistic Regression

3 Name: Afrida Islam

4 Student ID: 21221030

[ ]:

#Import packages

[1]:
```
import pandas as pd
#MinMaxScaler is for Normalization & StandardScaler is for Standardization
from sklearn.preprocessing import MinMaxScaler, StandardScaler
import numpy as np
import matplotlib.pyplot as plt
```

[2]:
```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

#Load Dataset from Google Drive

[ ]:
```
data = pd.read_csv("/content/drive/MyDrive/Colab Notebooks/
↪house_rent_data_log_reg.csv")
```

#Explore the Dataset

[ ]:
```
data.head()
```

[ ]:

	Size (sq. ft.)	Distance (miles)	House Rent ($)	Possibilty to purchase
0	498	11.9	675	0
1	513	8.6	750	0
2	621	8.3	800	1
3	710	3.4	965	0
4	650	7.5	855	1

[ ]:
```
data.iloc[3:6, 1:3]
```

[ ]:
```
   Distance (miles)  House Rent ($)
3               3.4             965
4               7.5             855
5               5.1             790
```

[ ]:
```
cols = data.columns
print(cols[1])
```

```
Distance (miles)
```

[ ]:
```
data[cols[-1]][6:10]
```

[ ]:
```
6    1
7    1
8    0
9    0
Name: Possibilty to purchase, dtype: int64
```

#Perform Feature Engineering

[ ]:
```
m = 3   # number of features
X = data.iloc[:, 0:m]
print(X)
```

```
   Size (sq. ft.)  Distance (miles)  House Rent ($)
0             498              11.9             675
1             513               8.6             750
2             621               8.3             800
3             710               3.4             965
4             650               7.5             855
5             620               5.1             790
6             560               4.6             740
7             780               8.1             810
8             450               5.4             710
9             790               4.5             940
```

[ ]:
```
y = data.iloc[:, -1]
print(y)
```

```
0    0
1    0
2    1
3    0
4    1
5    1
6    1
```

```
7    1
8    0
9    0
Name: Possibilty to purchase, dtype: int64
```

[ ]:
```
scaler = StandardScaler()
X_standard = scaler.fit_transform(X)
print(X_standard)
```

```
[[-1.09582295  2.10333853 -1.4389012 ]
 [-0.9602013   0.75818017 -0.5990756 ]
 [ 0.0162746   0.63589304 -0.03919186]
 [ 0.82096307 -1.36146331  1.80842447]
 [ 0.27847646  0.30979405  0.57668025]
 [ 0.00723315 -0.66850294 -0.15116861]
 [-0.53525345 -0.87231482 -0.71105234]
 [ 1.45386411  0.55436829  0.07278489]
 [-1.52981224 -0.54621582 -1.04698259]
 [ 1.54427855 -0.91307719  1.5284826 ]]
```

#Define the loss function

[ ]:
```
# sigmoid function
def sigmoid(x):
  return 1 / (1 + np.exp(-x))
```

[ ]:
```
sigmoid(0)
```

[ ]:
```
0.5
```

[ ]:
```
# Binary Cross Entropy
def bce(w0,w, X, y):
  y_hat_lin = np.add(np.dot(X, w), w0)
  y_hat = sigmoid(y_hat_lin) #logistic
  y_hat_log = np.log(y_hat) #logarithm

  loss = -(np.dot(y_hat_log, y)+ np.dot((1-y_hat_log), (1-y)))
  return loss
```

#Implement Gradient Descent to update model parameters

[ ]:
```
# Gradient descent
def gradient_descent(w0, w, X, y, alpha, epoch):
  train_loss = [0]* epoch

  for i in range(epoch):
    y_hat_lin = np.add(np.dot(X, w), w0)
    y_hat = sigmoid(y_hat_lin)
```

```
    err = y-y_hat

    gradient = -np.dot(X.T, err)

    w0 = w0 - alpha * (-np.sum(err))
    w = w - alpha * gradient

    train_loss[i] = bce(w0, w, X, y)
    print(f"Epoch - {i+1}: Training loss - {train_loss[i]}")

  return w0, w, train_loss
```

#Prepare the Dataset for training

[ ]:
```
n = int(len(y) * 0.8)   # number of training samples
X_train = X_standard[0:n, 0:m]
print(X_train)
X_test = X_standard[n: , 0:m]
print(X_test)
```

```
[[-1.09582295  2.10333853 -1.4389012 ]
 [-0.9602013   0.75818017 -0.5990756 ]
 [ 0.0162746   0.63589304 -0.03919186]
 [ 0.82096307 -1.36146331  1.80842447]
 [ 0.27847646  0.30979405  0.57668025]
 [ 0.00723315 -0.66850294 -0.15116861]
 [-0.53525345 -0.87231482 -0.71105234]
 [ 1.45386411  0.55436829  0.07278489]]
[[-1.52981224 -0.54621582 -1.04698259]
 [ 1.54427855 -0.91307719  1.5284826 ]]
```

[ ]:
```
y_train = y[0:n]
print(y_train)
y_test = y[n: ]
print(y_test)
```

```
0    0
1    0
2    1
3    0
4    1
5    1
6    1
7    1
Name: Possibilty to purchase, dtype: int64
8    0
9    0
Name: Possibilty to purchase, dtype: int64
```

#Train the model

[ ]:
```
w0 = 0
w = [0] * m

alpha = 0.001
epoch = 10000

updated_w0, updated_w, loss = gradient_descent(w0, w, X_train, y_train,  alpha,␣
 ↪epoch)
```

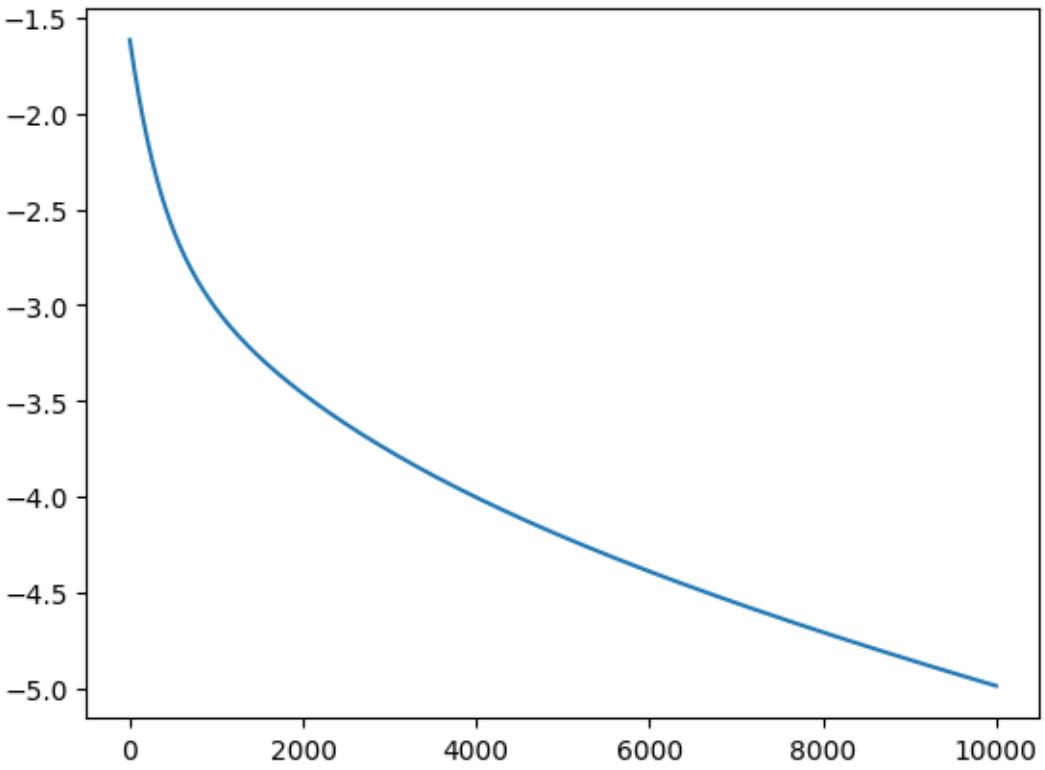
```
Streaming output truncated to the last 5000 lines.
Epoch - 5001: Training loss - -4.207433985896637
Epoch - 5002: Training loss - -4.2076255795057085
Epoch - 5003: Training loss - -4.207817149344784
Epoch - 5004: Training loss - -4.208008695422901
Epoch - 5005: Training loss - -4.208200217749095
Epoch - 5006: Training loss - -4.208391716332393
Epoch - 5007: Training loss - -4.208583191181822
Epoch - 5008: Training loss - -4.208774642306403
Epoch - 5009: Training loss - -4.208966069715152
Epoch - 5010: Training loss - -4.209157473417083
Epoch - 5011: Training loss - -4.209348853421204
Epoch - 5012: Training loss - -4.209540209736521
Epoch - 5013: Training loss - -4.209731542372032
Epoch - 5014: Training loss - -4.209922851336735
Epoch - 5015: Training loss - -4.210114136639624
Epoch - 5016: Training loss - -4.210305398289685
Epoch - 5017: Training loss - -4.2104966362958995
Epoch - 5018: Training loss - -4.210687850667252
Epoch - 5019: Training loss - -4.210879041412717
Epoch - 5020: Training loss - -4.211070208541262
Epoch - 5021: Training loss - -4.211261352061859
Epoch - 5022: Training loss - -4.2114524719834705
Epoch - 5023: Training loss - -4.211643568315054
Epoch - 5024: Training loss - -4.211834641065566
Epoch - 5025: Training loss - -4.212025690243959
Epoch - 5026: Training loss - -4.212216715859174
Epoch - 5027: Training loss - -4.212407717920161
Epoch - 5028: Training loss - -4.212598696435854
Epoch - 5029: Training loss - -4.212789651415188
Epoch - 5030: Training loss - -4.212980582867096
Epoch - 5031: Training loss - -4.2131714908005
Epoch - 5032: Training loss - -4.213362375224326
Epoch - 5033: Training loss - -4.21355323614749
Epoch - 5034: Training loss - -4.2137440735789085
Epoch - 5035: Training loss - -4.213934887527488
Epoch - 5036: Training loss - -4.214125678002136
```

```
Epoch - 9981: Training loss - -4.982084072564209
Epoch - 9982: Training loss - -4.982217465566897
Epoch - 9983: Training loss - -4.98235085290434
Epoch - 9984: Training loss - -4.982484234577624
Epoch - 9985: Training loss - -4.982617610587832
Epoch - 9986: Training loss - -4.98275098093605
Epoch - 9987: Training loss - -4.9828843456233605
Epoch - 9988: Training loss - -4.9830177046508455
Epoch - 9989: Training loss - -4.983151058019589
Epoch - 9990: Training loss - -4.983284405730674
Epoch - 9991: Training loss - -4.983417747785181
Epoch - 9992: Training loss - -4.983551084184194
Epoch - 9993: Training loss - -4.983684414928794
Epoch - 9994: Training loss - -4.983817740020058
Epoch - 9995: Training loss - -4.98395105945907
Epoch - 9996: Training loss - -4.984084373246908
Epoch - 9997: Training loss - -4.984217681384656
Epoch - 9998: Training loss - -4.984350983873387
Epoch - 9999: Training loss - -4.9844842807141845
Epoch - 10000: Training loss - -4.984617571908126
```

#Plot the loss in each epoch

[ ]:
```
plt.plot(loss)
```

[ ]:
```
[<matplotlib.lines.Line2D at 0x7aaa4a3e3b80>]
```



[ ]:
```
print(updated_w0)
print(updated_w)
```

```
0.9830524444417982
[ 3.48249062 -0.85393737 -2.75766437]
```

[3]:
```
data = pd.read_csv("/content/drive/MyDrive/Colab Notebooks/Rice_Cammeo_Osman.
↪csv")
```

[4]:
```
data.head()
```

[4]:

	Area	Perimeter	Major_Axis_Length	Minor_Axis_Length	Eccentricity	Convex_Area	Extent	Class
0	15231	525.578979	229.749878	85.093788	0.928882	15617	0.572896	1
1	14656	494.311005	206.020065	91.730972	0.895405	15072	0.615436	1
2	14634	501.122009	214.106781	87.768288	0.912118	14954	0.693259	1
3	13176	458.342987	193.337387	87.448395	0.891861			
4	14688	507.166992	211.743378	89.312454	0.906691			

```
3          13368  0.640669       1
4          15262  0.646024       1
```

[5]:
```
m = 7   # number of features
X = data.iloc[:, 0:m]
print(X)
```

```
       Area    Perimeter   Major_Axis_Length   Minor_Axis_Length  \
0     15231   525.578979          229.749878           85.093788
1     14656   494.311005          206.020065           91.730972
2     14634   501.122009          214.106781           87.768288
3     13176   458.342987          193.337387           87.448395
4     14688   507.166992          211.743378           89.312454
…      …          …                  …                   …
3805  11441   415.858002          170.486771           85.756592
3806  11625   421.390015          167.714798           89.462570
3807  12437   442.498993          183.572922           86.801979
3808   9882   392.296997          161.193985           78.210480
3809  11434   404.709992          161.079269           90.868195

       Eccentricity   Convex_Area    Extent
0          0.928882         15617  0.572896
1          0.895405         15072  0.615436
2          0.912118         14954  0.693259
3          0.891861         13368  0.640669
4          0.906691         15262  0.646024
…             …            …        …
3805       0.864280         11628  0.681012
3806       0.845850         11904  0.694279
3807       0.881144         12645  0.626739
3808       0.874406         10097  0.659064
3809       0.825692         11591  0.802949

[3810 rows x 7 columns]
```

[6]:
```
y = data.iloc[:, -1]
print(y)
```

```
0       1
1       1
2       1
3       1
4       1
       ..
3805    0
3806    0
3807    0
3808    0
```

```
3809    0
Name:  Class, Length: 3810, dtype: int64
```

[7]:
```
scaler = StandardScaler()
X_standard = scaler.fit_transform(X)
print(X_standard)
```

```
[[ 1.47982953  2.0043543   2.34854657 …  2.01833746  1.49965944
  -1.15292093]
 [ 1.14787029  1.12585309  0.98839042 …  0.41001813  1.19291767
  -0.60207876]
 [ 1.13516924  1.31721425  1.45190846 …  1.21295648  1.12650386
   0.405611  ]
 …
 [-0.13320373 -0.32985087 -0.29824512 … -0.27509915 -0.17306812
  -0.45573108]
 [-1.60825742 -1.74032002 -1.58097116 … -0.59882135 -1.60715621
  -0.03716757]
 [-0.71225612 -1.39156604 -1.58754648 … -2.93916012 -0.76628981
   1.82594693]]
```

[8]:
```
# sigmoid function
def sigmoid(x):
  return 1 / (1 + np.exp(-x))
```

[9]:
```
# Binary Cross Entropy
def bce(w0,w, X, y):
  y_hat_lin = np.add(np.dot(X, w), w0)
  y_hat = sigmoid(y_hat_lin) #logistic
  y_hat_log = np.log(y_hat) #logarithm

  loss = -(np.dot(y_hat_log, y)+ np.dot((1-y_hat_log), (1-y)))
  return loss
```

[10]:
```
# Gradient descent
def gradient_descent(w0, w, X, y, alpha, epoch):
  train_loss = [0]* epoch

  for i in range(epoch):
    y_hat_lin = np.add(np.dot(X, w), w0)
    y_hat = sigmoid(y_hat_lin)
    err = y-y_hat

    gradient = -np.dot(X.T, err)

    w0 = w0 - alpha * (-np.sum(err))
    w = w - alpha * gradient
```

```
    train_loss[i] = bce(w0, w, X, y)
    print(f"Epoch - {i+1}: Training loss - {train_loss[i]}")

  return w0, w, train_loss
```

[11]:
```
n = int(len(y) * 0.8)   # number of training samples
X_train = X_standard[0:n, 0:m]
print(X_train)
X_test = X_standard[n: , 0:m]
print(X_test)
```

```
[[ 1.47982953  2.0043543   2.34854657 …  2.01833746  1.49965944
  -1.15292093]
 [ 1.14787029  1.12585309  0.98839042 …  0.41001813  1.19291767
  -0.60207876]
 [ 1.13516924  1.31721425  1.45190846 …  1.21295648  1.12650386
   0.405611  ]
 …
 [-1.49221601 -1.44152009 -1.13200576 …  0.16304503 -1.50247004
  -0.2243131 ]
 [ 0.06481717 -0.48353536 -0.78293572 … -2.0139966  -0.00703359
   0.57462398]
 [ 0.07751822  0.09613933  0.18839148 …  0.40691406  0.02729898
   1.22256639]]
[[-1.20009188 -0.40826605  0.16786614 …  1.97639208 -1.20191939
  -1.95498146]
 [-0.67299833 -0.68826966 -0.60255873 … -0.13189842 -0.69649902
   1.44601405]
 [-0.14590478 -0.42827047 -0.4394798  … -0.65208636 -0.20627503
  -0.78099929]
 …
 [-0.13320373 -0.32985087 -0.29824512 … -0.27509915 -0.17306812
  -0.45573108]
 [-1.60825742 -1.74032002 -1.58097116 … -0.59882135 -1.60715621
  -0.03716757]
 [-0.71225612 -1.39156604 -1.58754648 … -2.93916012 -0.76628981
   1.82594693]]
```

[12]:
```
y_train = y[0:n]
print(y_train)
y_test = y[n: ]
print(y_test)
```

```
0       1
1       1
2       1
```

```
3       1
4       1
       ..
3043    0
3044    0
3045    0
3046    0
3047    0
Name:  Class, Length: 3048, dtype: int64
3048    0
3049    0
3050    0
3051    0
3052    0
       ..
3805    0
3806    0
3807    0
3808    0
3809    0
Name:  Class, Length: 762, dtype: int64
```

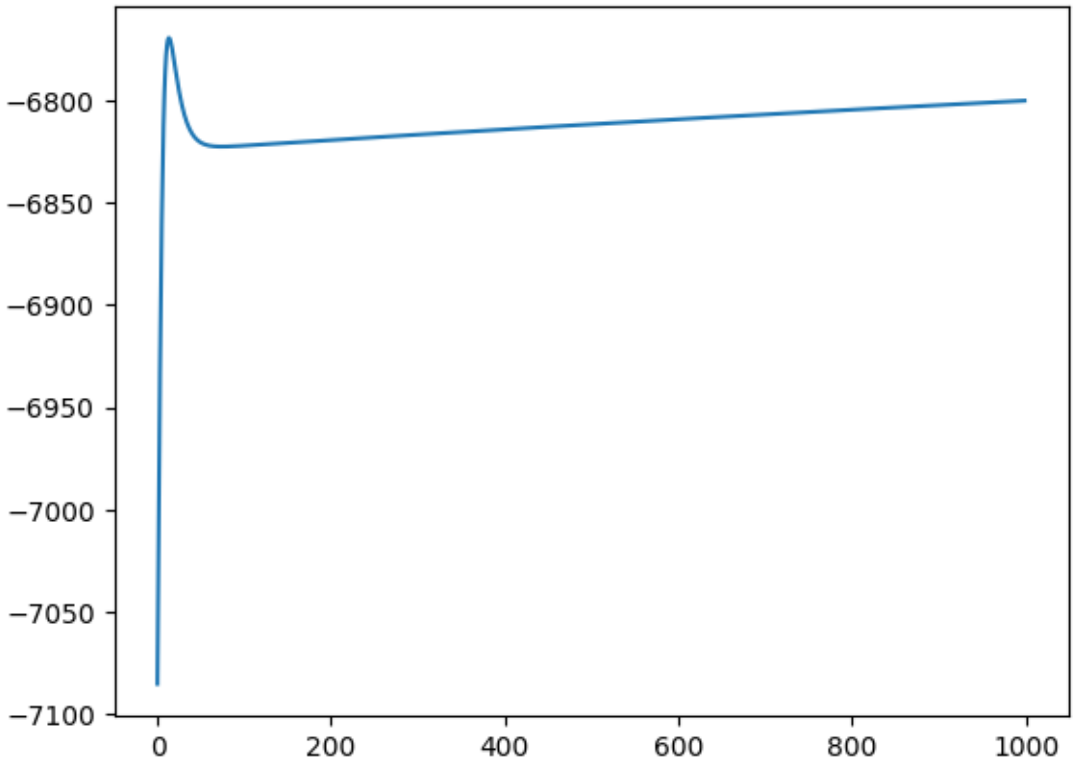
[26]:
```
w0 = 0
w = [0] * m

alpha = 0.001
epoch = 1000


updated_w0, updated_w, loss = gradient_descent(w0, w, X_train, y_train,  alpha,␣
 ↪epoch)
```

```
Epoch - 1: Training loss - -7085.123568728295
Epoch - 2: Training loss - -7029.884965773944
Epoch - 3: Training loss - -6977.566729160317
Epoch - 4: Training loss - -6931.047876789875
Epoch - 5: Training loss - -6891.387807400673
Epoch - 6: Training loss - -6858.680661065756
Epoch - 7: Training loss - -6832.5015860635285
Epoch - 8: Training loss - -6812.160166241046
Epoch - 9: Training loss - -6796.855427471239
Epoch - 10: Training loss - -6785.773189726914
Epoch - 11: Training loss - -6778.145283849048
Epoch - 12: Training loss - -6773.282106007351
Epoch - 13: Training loss - -6770.586667531899
Epoch - 14: Training loss - -6769.556431869829
Epoch - 15: Training loss - -6769.777750471592
Epoch - 16: Training loss - -6770.916396013069
```

```
Epoch - 977: Training loss - -6800.728911277302
Epoch - 978: Training loss - -6800.707120454835
Epoch - 979: Training loss - -6800.685334703136
Epoch - 980: Training loss - -6800.6635540191755
Epoch - 981: Training loss - -6800.641778399913
Epoch - 982: Training loss - -6800.620007842332
Epoch - 983: Training loss - -6800.598242343408
Epoch - 984: Training loss - -6800.576481900119
Epoch - 985: Training loss - -6800.554726509448
Epoch - 986: Training loss - -6800.532976168387
Epoch - 987: Training loss - -6800.511230873921
Epoch - 988: Training loss - -6800.489490623045
Epoch - 989: Training loss - -6800.467755412756
Epoch - 990: Training loss - -6800.446025240057
Epoch - 991: Training loss - -6800.424300101949
Epoch - 992: Training loss - -6800.402579995439
Epoch - 993: Training loss - -6800.38086491754
Epoch - 994: Training loss - -6800.35915486526
Epoch - 995: Training loss - -6800.337449835622
Epoch - 996: Training loss - -6800.3157498256405
Epoch - 997: Training loss - -6800.294054832341
Epoch - 998: Training loss - -6800.272364852757
Epoch - 999: Training loss - -6800.250679883908
Epoch - 1000: Training loss - -6800.22899992283
```

[27]:
```
plt.plot(loss)
```

[27]:
```
[<matplotlib.lines.Line2D at 0x7b79ca267a00>]
```



[28]:
```
print(updated_w0)
print(updated_w)
```

```
-0.3027361096625728
[ 0.44613794  1.006069    0.53811413 -0.49905008  1.32479183  2.41981724
  0.04347242]
```

[ ]:
```
# Install the package for Tex and then convert to PDF directly as LaTex
!sudo apt-get install texlive-xetex texlive-fonts-recommended␣
  ↪texlive-plain-generic pandoc

# Provide the file path of the notebook file
!jupyter nbconvert --to pdf "/content/drive/MyDrive/Colab Notebooks/
  ↪21221030_Lab_4.ipynb"
```

```
Reading package lists… Done
Building dependency tree… Done
Reading state information… Done
The following additional packages will be installed:
  dvisvgm fonts-droid-fallback fonts-lato fonts-lmodern fonts-noto-mono
  fonts-texgyre fonts-urw-base35 libapache-pom-java
  libcmark-gfm-extensions0.29.0.gfm.3 libcmark-gfm0.29.0.gfm.3
```