TRIBHUVAN UNIVERSITY
INSTITUTE OF SCIENCE AND TECHNOLOGY

Central Department of Computer Science and Information Technology

Kirtipur, Kathmandu

2022

Lab Report: I

Machine Learning

Submitted By:

Sudhan Kandel

Semester: 1st

Roll no: 2

Submitted To:

MR. Arjun Singh Saud

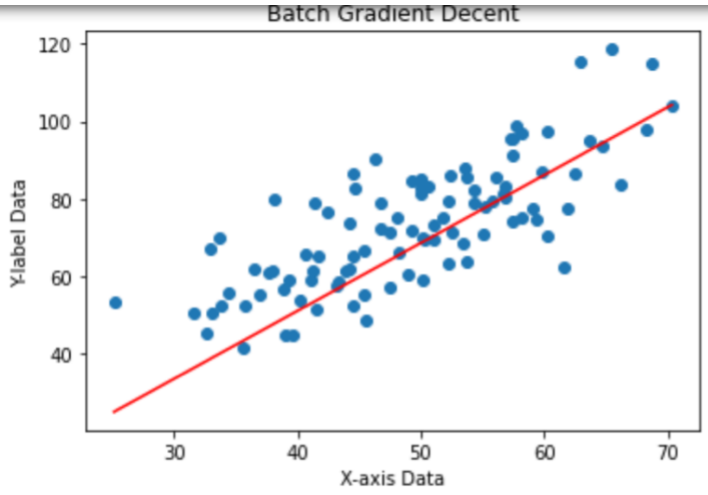
_____

1. Write python programs to implement linear regression using Stochastic GD

```
import pandas as pd
import matplotlib.pyplot as plt
from math import sqrt
from sklearn.metrics import mean_squared_error
from datetime import datetime as dt
df=pd.read_csv('data.csv')
x=df.iloc[:,0]
y=df.iloc[:,1]
w0=w1=0
lr=0.0001
epoch=100
n=len(x)
start = dt.now()
for i in range(epoch):
    yp=w0+w1*x
    w0=w0+lr*(1/n)*sum((y-yp))
    w1=w1+lr*(1/n)*sum((y-yp)*x)
    print("Updated Weights values are ",w0,w1)
running_secs = (dt.now() - start).microseconds
yp=w1*x+w0
plt.scatter(x,y)
plt.plot([min(x),max(x)],[min(x),max(yp)],color='red')
plt.title("Batch Gradient Decent")
plt.xlabel('X-axis Data')
plt.ylabel('Y-label Data')
plt.show()
rmse=sqrt(mean_squared_error(y,yp))
print("\nThis model takes ",running_secs," Microseconds")
print("Root Mean Square Error is ",rmse)
```

Output:

```
Updated Weights values are  0.007314947524769776 0.37121676427077466
Updated Weights values are  0.012805577398524209 0.6493910275598956
Updated Weights values are  0.016929139279147676 0.8578431008009637
Updated Weights values are  0.020028277089690148 1.0140482574846188
Updated Weights values are  0.02235975397032826 1.1311017425431509
Updated Weights values are  0.024115977565611887 1.2188165892277145
Updated Weights values are  0.025441130181676472 1.2845462775294185
Updated Weights values are  0.026443256056843053 1.3338012002692872
Updated Weights values are  0.027203319060724127 1.37071061720987
Updated Weights values are  0.02778199015814389 1.3983688243007324
Updated Weights values are  0.028224733652998865 1.4190945585490093
Updated Weights values are  0.02856561860838681 1.4346253852670985
Updated Weights values are  0.0288301749667215 1.4462633638112603
Updated Weights values are  0.029037533807650805 1.4549842034426828
Updated Weights values are  0.029202031168544574 1.4615190602010584
Updated Weights values are  0.02933440988102827 1.4664158320880194
Updated Weights values are  0.029442720177418953 1.4700850911345629
Updated Weights values are  0.02953299456782369 1.472834502986731
Updated Weights values are  0.029609753565037485 1.4748946195151038
```




```
This model takes  47973  Microseconds
Root Mean Square Error is  10.538538374381778
```

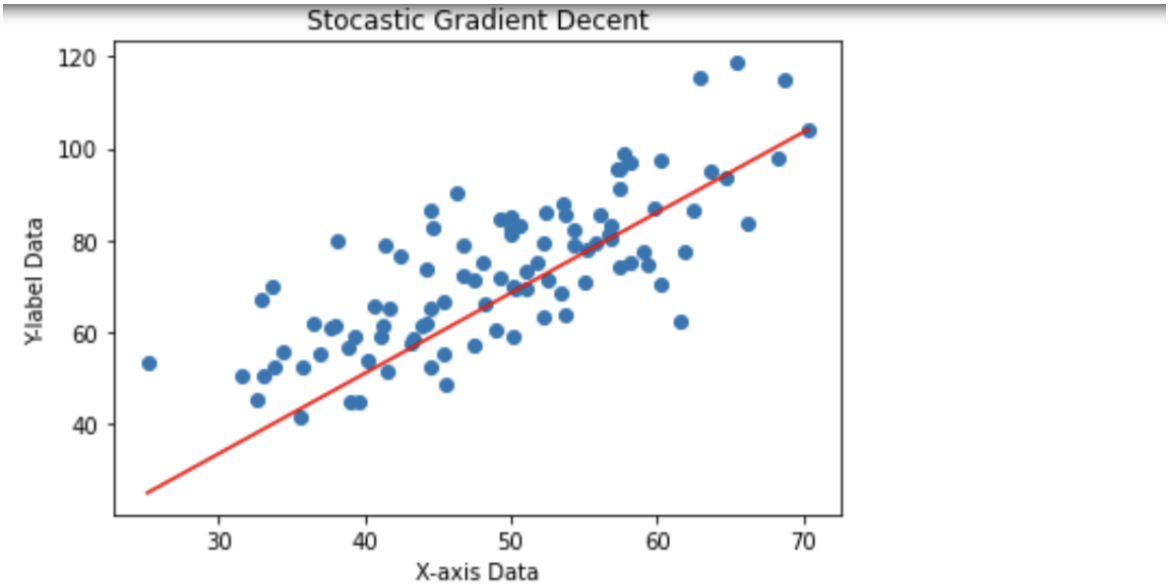
2. Write python programs to implement linear regression using Batch GD.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from math import sqrt
from sklearn.metrics import mean_squared_error
from datetime import datetime as dt
df=pd.read_csv('data.csv')
x=df.iloc[:,0]
y=df.iloc[:,1]
w0=w1=0
lr=0.0001
epoch=100
n=len(x)
start = dt.now()
for i in range(epoch):
    np.random.shuffle([x,y])
    for i in range(len(x)):
        yp=w0+w1*x
        w0=w0+lr*(1/n)*sum((y-yp))
        w1=w1+lr*(1/n)*sum((y-yp)*x)
        print("Updated Weights values are ",w0,w1)
```

```
running_secs = (dt.now() - start).microseconds

yp=w1*x+w0

plt.scatter(x,y)

plt.plot([min(x),max(x)],[min(x),max(yp)],color='red')

plt.title("Stocastic Gradient Decent")

plt.xlabel('X-axis Data')

plt.ylabel('Y-label Data')

plt.show()

rmse=sqrt(mean_squared_error(y,yp))

print("\nThis model takes ",running_secs," Microseconds")

print("Root Mean Square Error is ",rmse)
```

Output:

```
Updated Weights values are  0.38180232803710085 1.4741528127973422
Updated Weights values are  0.38183738396154815 1.4741521254433225
Updated Weights values are  0.38187243975699964 1.4741514380918321
Updated Weights values are  0.3819074954234558 1.474150750742871
Updated Weights values are  0.3819425509609171 1.474150063396439
Updated Weights values are  0.381977606369384 1.4741493760525364
Updated Weights values are  0.38201266164885694 1.474148688711163
Updated Weights values are  0.38204771679933647 1.4741480013723187
Updated Weights values are  0.382082771820823 1.4741473140360037
Updated Weights values are  0.38211782671331707 1.474146626702218
Updated Weights values are  0.38215288147681914 1.4741459393709615
Updated Weights values are  0.3821879361113296 1.474145252042234
Updated Weights values are  0.38222299061684906 1.4741445647160358
Updated Weights values are  0.3822580449933779 1.4741438773923667
Updated Weights values are  0.3822930992409166 1.4741431900712267
Updated Weights values are  0.3823281533594657 1.474142502752616
Updated Weights values are  0.38236320734902557 1.4741418154365344
Updated Weights values are  0.38239826120959675 1.474141128122982
Updated Weights values are  0.3824333149411797 1.4741404408119585
Updated Weights values are  0.382468368543775 1.4741397535034642
```




```
This model takes  282088  Microseconds
Root Mean Square Error is  10.526678630184813
```

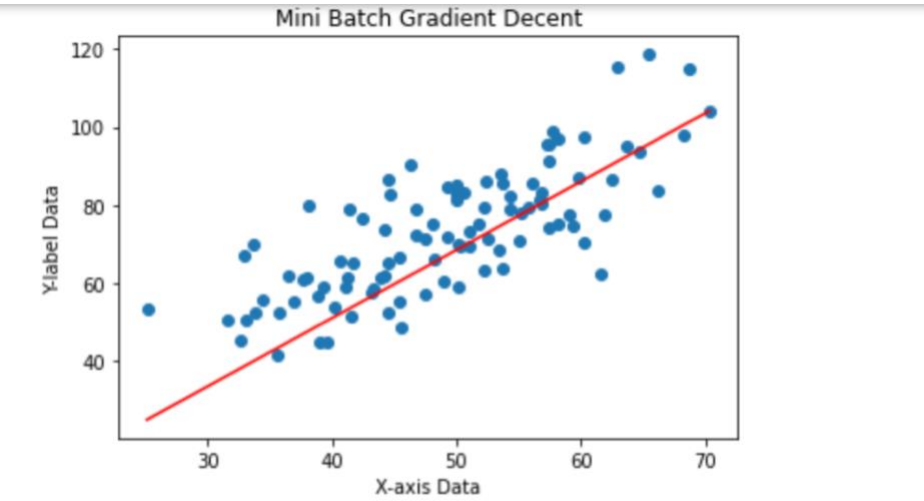
3. Write python programs to implement linear regression using mini-batch GD (use batch size 8).

```
import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

from math import sqrt

from sklearn.metrics import mean_squared_error

from datetime import datetime as dt

df=pd.read_csv('data.csv')

x=df.iloc[:,0]

y=df.iloc[:,1]

w0=w1=0

lr=0.0001

epoch=100

n=len(x)

b_size=8

start = dt.now()

for i in range(epoch):

    print("epoch.....",i)

    np.random.shuffle([x,y])

    for i in range(b_size):

        print("Batch.....",i)

        for j in range(13):
```

```
        yp=w0+w1*x

        w0=w0+lr*(1/n)*sum((y-yp))

        w1=w1+lr*(1/n)*sum((y-yp)*x)

        print("Updated Weights values are ",w0,w1)
running_secs = (dt.now() - start).microseconds

yp=w1*x+w0

plt.scatter(x,y)

plt.plot([min(x),max(x)],[min(x),max(yp)],color='red')

plt.title("Mini Batch Gradient Decent")

plt.xlabel('X-axis Data')

plt.ylabel('Y-label Data')

plt.show()

rmse=sqrt(mean_squared_error(y,yp))

print("\nThis model takes ",running_secs," Microseconds")

print("Root Mean Square Error is ",rmse)
```

Output:

```
Updated Weights values are  0.3993491993435807 1.4738087650303695
Updated Weights values are  0.39938419070053993 1.4738080789423476
Updated Weights values are  0.39941918192874093 1.4738073928568503
Updated Weights values are  0.3994541730281842 1.4738067067738776
Updated Weights values are  0.3994891639988702 1.4738060206934294
Updated Weights values are  0.39952415484079934 1.4738053346155058
Batch..... 7
Updated Weights values are  0.39955914555397215 1.4738046485401068
Updated Weights values are  0.3995941361383891 1.4738039624672326
Updated Weights values are  0.3996291265940507 1.4738032763968827
Updated Weights values are  0.3996641169209574 1.4738025903290575
Updated Weights values are  0.3996991071191096 1.4738019042637567
Updated Weights values are  0.3997340971885079 1.4738012182009805
Updated Weights values are  0.39976908712915266 1.4738005321407288
Updated Weights values are  0.3998040769410444 1.4737998460830015
Updated Weights values are  0.39983906662418367 1.4737991600277989
Updated Weights values are  0.39987405617857086 1.4737984739751206
Updated Weights values are  0.3999090456042064 1.4737977879249669
Updated Weights values are  0.3999440349010909 1.4737971018773377
Updated Weights values are  0.3999790240692247 1.4737964158322328
```




```
This model takes  597755  Microseconds
Root Mean Square Error is  10.526095827266413
```

Code Link: https://github.com/sudhankandel/Machine-Learning-Lab