```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.datasets import make_classification
from sklearn.linear_model import LogisticRegression, SGDClassifier
from mlxtend.plotting import plot_decision_regions
from sklearn.utils import shuffle
```

```
!pip install --upgrade --no-cache-dir gdown
!gdown 1Won6xkyYCcJLJ7eMpVt5VA_4P0tE1nb7
```

```
Requirement already satisfied: gdown in /usr/local/lib/python3.10/dist-packages (4.7.3)
Collecting gdown
  Downloading gdown-5.0.0-py3-none-any.whl (16 kB)
Requirement already satisfied: beautifulsoup4 in /usr/local/lib/python3.10/dist-packages (from gdown) (4.11.2)
Requirement already satisfied: filelock in /usr/local/lib/python3.10/dist-packages (from gdown) (3.13.1)
Requirement already satisfied: requests[socks] in /usr/local/lib/python3.10/dist-packages (from gdown) (2.31.0)
Requirement already satisfied: tqdm in /usr/local/lib/python3.10/dist-packages (from gdown) (4.66.1)
Requirement already satisfied: soupsieve>1.2 in /usr/local/lib/python3.10/dist-packages (from beautifulsoup4->gdown) (2.5)
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.10/dist-packages (from requests[socks]->gdown) (3.3.2)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests[socks]->gdown) (3.6)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests[socks]->gdown) (2.0.7)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from requests[socks]->gdown) (2023.11.17)
Requirement already satisfied: PySocks!=1.5.7,>=1.5.6 in /usr/local/lib/python3.10/dist-packages (from requests[socks]->gdown) (1.7.1)
Installing collected packages: gdown
  Attempting uninstall: gdown
    Found existing installation: gdown 4.7.3
    Uninstalling gdown-4.7.3:
      Successfully uninstalled gdown-4.7.3
Successfully installed gdown-5.0.0
Downloading...
From: https://drive.google.com/uc?id=1Won6xkyYCcJLJ7eMpVt5VA_4P0tE1nb7
To: /content/data_banknote_authentication.txt
100% 46.4k/46.4k [00:00<00:00, 78.9MB/s]
```

```
df = pd.read_csv('/content/data_banknote_authentication.txt')
df
```

	x1	x2	x3	x4	y
0	3.62160	8.66610	-2.8073	-0.44699	0
1	4.54590	8.16740	-2.4586	-1.46210	0
2	3.86600	-2.63830	1.9242	0.10645	0
3	3.45660	9.52280	-4.0112	-3.59440	0
4	0.32924	-4.45520	4.5718	-0.98880	0
...	...	...	...	...	...
1367	0.40614	1.34920	-1.4501	-0.55949	1
1368	-1.38870	-4.87730	6.4774	0.34179	1
1369	-3.75030	-13.45860	17.5932	-2.77710	1
1370	-3.56370	-8.38270	12.3930	-1.28230	1
1371	-2.54190	-0.65804	2.6842	1.19520	1

1372 rows × 5 columns

```
shuffled_data = shuffle(df)
shuffled_data.to_csv('created_data.csv', index=False)
print(shuffled_data)
```

```
            x1       x2        x3       x4  y
638   2.7213   7.0500  -0.58808  0.41809  0
67    2.4235   9.5332  -3.07890 -2.77460  0
531   3.3583  10.3567  -3.73010 -3.69910  0
1171 -3.8552   3.5219  -0.38415 -3.86080  1
386  -2.4953  11.1472   1.93530 -3.46380  0
...      ...      ...       ...      ... ..
289   3.2422   6.2265   0.12224 -1.44660  0
701   5.5910  10.4643  -4.38390 -4.33790  0
```

```
1192 -4.4018 -12.9371  15.65590 -1.68060  1
640   4.1665  -0.4449   0.23448  0.27843  0
429   2.5503  -4.9518   6.37290 -0.41596  0

[1372 rows x 5 columns]
```

```
df2 = pd.read_csv('/content/created_data.csv')
df2
```

	x1	x2	x3	x4	y
0	2.7213	7.0500	-0.58808	0.41809	0
1	2.4235	9.5332	-3.07890	-2.77460	0
2	3.3583	10.3567	-3.73010	-3.69910	0
3	-3.8552	3.5219	-0.38415	-3.86080	1
4	-2.4953	11.1472	1.93530	-3.46380	0
...	...	...	...	...	...
1367	3.2422	6.2265	0.12224	-1.44660	0
1368	5.5910	10.4643	-4.38390	-4.33790	0
1369	-4.4018	-12.9371	15.65590	-1.68060	1
1370	4.1665	-0.4449	0.23448	0.27843	0
1371	2.5503	-4.9518	6.37290	-0.41596	0

1372 rows × 5 columns

```
class_counts = df2['y'].value_counts()

# نمایش تعداد نمونه‌ها برای هر کلاس
print(class_counts)
```

```
0    762
1    610
Name: y, dtype: int64
```

```
class_0_samples = df2[df2['y'] == 0].sample(610)
class_1_samples = df2[df2['y'] == 1].sample(610)

# ایجاد دیتافریم جدید با 1220 نمونه انتخابی
df1 = pd.concat([class_0_samples, class_1_samples], ignore_index=True)
```

Logistic Regression (from Scratch)

```
def sigmoid(x):
    return 1 / (1 + np.exp(-x))
```

```
def logistic_regression(x, w):
    y_hat = sigmoid(x @ w)
    return y_hat
```

Binary Cross Entropy (BCE)

```
def bce(y, y_hat):
    loss = -(np.mean(y*np.log(y_hat) + (1-y)*np.log(1-y_hat)))
    return loss
```

Gradient

```
def gradient(x, y, y_hat):
    grads = (x.T @ (y_hat - y)) / len(y)
    return grads
```

Gradient Descent

```
def gradient_descent(w, eta, grads):
    w -= eta*grads
    return w
```

Accuracy

```
def accuracy(y, y_hat):
    acc = np.sum(y == np.round(y_hat)) / len(y)
    return acc
```

Train

```
X = df1[['x1','x2','x3','x4']].values
y = df1[['y']].values
X, y
```

```
(array([[ 4.9249 ,  0.68906,  0.77344,  1.2095 ],
        [ 4.0948 , -2.9674 ,  2.3689 ,  0.75429],
        [ 2.0153 ,  0.43661,  4.5864 , -0.3151 ],
        ...,
        [-4.9462 ,  3.5716 ,  0.82742, -1.4957 ],
        [-2.588  ,  3.8654 , -0.3336 , -1.2797 ],
        [-3.5741 ,  3.944  , -0.07912, -2.1203 ]]),
 array([[0],
        [0],
        [0],
        ...,
        [1],
        [1],
        [1]]))
```

```
x_train, x_test, y_train, y_test = train_test_split(X, y, test_size=0.2,random_state=42)
```

```
x_train.shape, x_test.shape, y_train.shape, y_test.shape
```

```
((976, 4), (244, 4), (976, 1), (244, 1))
```

```
y_hat = logistic_regression(x_train, np.random.randn(4, 1))
print(y_hat.shape)
```

```
(976, 1)
```

```
x_train = np.hstack((np.ones((len(x_train), 1)), x_train))
x_train.shape
```

```
(976, 5)
```

```
m = 4
w = np.random.randn(m+1, 1)
print(w.shape)
```

```
(5, 1)
```

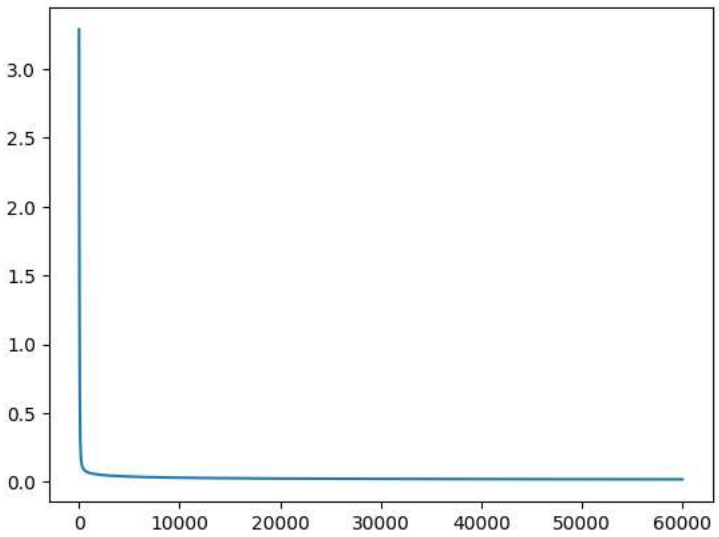
```
eta = 0.01
n_epochs = 60000 #N
```

```
error_hist = []

for epoch in range(n_epochs):
    # predictions
    y_hat = logistic_regression(x_train, w)

    # loss
    e = bce(y_train, y_hat)
    error_hist.append(e)

    # gradients
    grads = gradient(x_train, y_train, y_hat)

    # gradient descent
    w = gradient_descent(w, eta, grads)

    if (epoch+1) %  1== 0:
        print(f'Epoch={epoch}, \t E={e:.4},\t w={w.T[0]}')
```

```
 Epoch=59942,     E=0.01938,      w=[ 4.00398546 -3.4662217  -2.02849042 -2.46122729 -0.1578695 ]
 Epoch=59943,     E=0.01938,      w=[ 4.00400029 -3.46623788 -2.02849892 -2.46123841 -0.15787025]
 Epoch=59944,     E=0.01938,      w=[ 4.00401513 -3.46625406 -2.02850742 -2.46124953 -0.15787099]
 Epoch=59945,     E=0.01938,      w=[ 4.00402996 -3.46627023 -2.02851593 -2.46126065 -0.15787174]
 Epoch=59946,     E=0.01938,      w=[ 4.0040448  -3.46628641 -2.02852443 -2.46127177 -0.15787248]
 Epoch=59947,     E=0.01938,      w=[ 4.00405963 -3.46630259 -2.02853294 -2.46128289 -0.15787323]
 Epoch=59948,     E=0.01938,      w=[ 4.00407447 -3.46631877 -2.02854144 -2.461294   -0.15787397]
 Epoch=59949,     E=0.01938,      w=[ 4.0040893  -3.46633495 -2.02854994 -2.46130512 -0.15787472]
 Epoch=59950,     E=0.01938,      w=[ 4.00410414 -3.46635112 -2.02855845 -2.46131624 -0.15787546]
 Epoch=59951,     E=0.01938,      w=[ 4.00411897 -3.4663673  -2.02856695 -2.46132736 -0.15787621]
 Epoch=59952,     E=0.01938,      w=[ 4.0041338  -3.46638348 -2.02857545 -2.46133847 -0.15787695]
 Epoch=59953,     E=0.01938,      w=[ 4.00414864 -3.46639965 -2.02858396 -2.46134959 -0.1578777 ]
 Epoch=59954,     E=0.01938,      w=[ 4.00416347 -3.46641583 -2.02859246 -2.46136071 -0.15787844]
 Epoch=59955,     E=0.01938,      w=[ 4.0041783  -3.46643201 -2.02860096 -2.46137182 -0.15787918]
 Epoch=59956,     E=0.01938,      w=[ 4.00419313 -3.46644818 -2.02860947 -2.46138294 -0.15787993]
 Epoch=59957,     E=0.01938,      w=[ 4.00420797 -3.46646436 -2.02861797 -2.46139406 -0.15788067]
 Epoch=59958,     E=0.01938,      w=[ 4.0042228  -3.46648053 -2.02862647 -2.46140517 -0.15788142]
 Epoch=59959,     E=0.01938,      w=[ 4.00423763 -3.46649671 -2.02863497 -2.46141629 -0.15788216]
 Epoch=59960,     E=0.01938,      w=[ 4.00425246 -3.46651289 -2.02864348 -2.46142741 -0.15788291]
 Epoch=59961,     E=0.01938,      w=[ 4.00426729 -3.46652906 -2.02865198 -2.46143852 -0.15788365]
 Epoch=59962,     E=0.01938,      w=[ 4.00428213 -3.46654524 -2.02866048 -2.46144964 -0.1578844 ]
 Epoch=59963,     E=0.01938,      w=[ 4.00429696 -3.46656141 -2.02866898 -2.46146075 -0.15788514]
 Epoch=59964,     E=0.01938,      w=[ 4.00431179 -3.46657759 -2.02867749 -2.46147187 -0.15788589]
 Epoch=59965,     E=0.01938,      w=[ 4.00432662 -3.46659376 -2.02868599 -2.46148299 -0.15788663]
 Epoch=59966,     E=0.01938,      w=[ 4.00434145 -3.46660993 -2.02869449 -2.4614941  -0.15788738]
 Epoch=59967,     E=0.01938,      w=[ 4.00435628 -3.46662611 -2.02870299 -2.46150522 -0.15788812]
 Epoch=59968,     E=0.01938,      w=[ 4.00437111 -3.46664228 -2.02871149 -2.46151633 -0.15788887]
 Epoch=59969,     E=0.01938,      w=[ 4.00438594 -3.46665846 -2.02871999 -2.46152745 -0.15788961]
 Epoch=59970,     E=0.01938,      w=[ 4.00440077 -3.46667463 -2.0287285  -2.46153856 -0.15789036]
 Epoch=59971,     E=0.01938,      w=[ 4.0044156  -3.4666908  -2.028737   -2.46154968 -0.1578911 ]
 Epoch=59972,     E=0.01938,      w=[ 4.00443043 -3.46670698 -2.0287455  -2.46156079 -0.15789185]
 Epoch=59973,     E=0.01938,      w=[ 4.00444526 -3.46672315 -2.028754   -2.4615719  -0.15789259]
 Epoch=59974,     E=0.01938,      w=[ 4.00446009 -3.46673932 -2.0287625  -2.46158302 -0.15789334]
 Epoch=59975,     E=0.01938,      w=[ 4.00447492 -3.4667555  -2.028771   -2.46159413 -0.15789408]
 Epoch=59976,     E=0.01938,      w=[ 4.00448974 -3.46677167 -2.0287795  -2.46160525 -0.15789483]
 Epoch=59977,     E=0.01938,      w=[ 4.00450457 -3.46678784 -2.028788   -2.46161636 -0.15789557]
 Epoch=59978,     E=0.01938,      w=[ 4.0045194  -3.46680401 -2.0287965  -2.46162747 -0.15789632]
 Epoch=59979,     E=0.01938,      w=[ 4.00453423 -3.46682018 -2.028805   -2.46163859 -0.15789706]
 Epoch=59980,     E=0.01938,      w=[ 4.00454906 -3.46683636 -2.0288135  -2.4616497  -0.15789781]
 Epoch=59981,     E=0.01938,      w=[ 4.00456388 -3.46685253 -2.028822   -2.46166081 -0.15789855]
 Epoch=59982,     E=0.01938,      w=[ 4.00457871 -3.4668687  -2.0288305  -2.46167193 -0.1578993 ]
 Epoch=59983,     E=0.01938,      w=[ 4.00459354 -3.46688487 -2.028839   -2.46168304 -0.15790004]
 Epoch=59984,     E=0.01938,      w=[ 4.00460837 -3.46690104 -2.0288475  -2.46169415 -0.15790079]
 Epoch=59985,     E=0.01938,      w=[ 4.00462319 -3.46691721 -2.028856   -2.46170527 -0.15790153]
 Epoch=59986,     E=0.01938,      w=[ 4.00463802 -3.46693338 -2.0288645  -2.46171638 -0.15790227]
 Epoch=59987,     E=0.01938,      w=[ 4.00465285 -3.46694955 -2.028873   -2.46172749 -0.15790302]
 Epoch=59988,     E=0.01938,      w=[ 4.00466767 -3.46696572 -2.0288815  -2.4617386  -0.15790376]
 Epoch=59989,     E=0.01938,      w=[ 4.0046825  -3.46698189 -2.02889    -2.46174972 -0.15790451]
 Epoch=59990,     E=0.01938,      w=[ 4.00469732 -3.46699806 -2.0288985  -2.46176083 -0.15790525]
 Epoch=59991,     E=0.01938,      w=[ 4.00471215 -3.46701423 -2.028907   -2.46177194 -0.157906  ]
 Epoch=59992,     E=0.01938,      w=[ 4.00472697 -3.4670304  -2.0289155  -2.46178305 -0.15790674]
 Epoch=59993,     E=0.01938,      w=[ 4.0047418  -3.46704657 -2.028924   -2.46179416 -0.15790749]
 Epoch=59994,     E=0.01938,      w=[ 4.00475663 -3.46706274 -2.0289325  -2.46180528 -0.15790823]
 Epoch=59995,     E=0.01938,      w=[ 4.00477145 -3.46707891 -2.028941   -2.46181639 -0.15790898]
 Epoch=59996,     E=0.01938,      w=[ 4.00478627 -3.46709508 -2.02894949 -2.4618275  -0.15790972]
 Epoch=59997,     E=0.01938,      w=[ 4.0048011  -3.46711125 -2.02895799 -2.46183861 -0.15791047]
 Epoch=59998,     E=0.01938,      w=[ 4.00481592 -3.46712742 -2.02896649 -2.46184972 -0.15791121]
 Epoch=59999,     E=0.01938,      w=[ 4.00483075 -3.46714358 -2.02897499 -2.46186083 -0.15791196]
```

```
plt.plot(error_hist)
```

```
[<matplotlib.lines.Line2D at 0x791b1bfd9840>]
```




```
y_hat = logistic_regression(x_train, w)
accuracy(y_train, y_hat)
```

```
0.9938524590163934
```

```
x_test = np.hstack((np.ones((len(x_test), 1)), x_test))
y_hat = logistic_regression(x_test, w)
accuracy(y_test, y_hat)
```

```
0.9877049180327869
```