```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

```
class Perceptron:

    def __init__(self, dim):
        # Initialize weights randomly
        self.weights = np.random.randn(dim + 1)

    def predict(self, x):
        # Add bias term to input
        if (x.shape[0] < 3):
          x = np.hstack((1, x))

        # Calculate dot product of weights and input
        #print('error')
        #print(self.weights)
        #print(x)
        dot_product = np.dot(self.weights, x)

        # Return predicted class label
        return 1 if dot_product > 0 else 0

    def update_weights(self, x, y):
        # Add bias term to input
        x = np.hstack((1, x))

        # Calculate prediction and error
        prediction = self.predict(x)
        error = y - prediction

        # Update weights
        self.weights += error * x

def train_perceptron(X, y, num_epochs):
    # Initialize perceptron
    input_dim = X.shape[1]
    perceptron = Perceptron(input_dim)

    # Train perceptron
    num_iterations = 0
    for epoch in range(num_epochs):
```

```
        for i in range(len(X)):
            perceptron.update_weights(X[i], y[i])
            num_iterations += 1

  # Return trained perceptron and number of iterations
    return perceptron, num_iterations

def test_perceptron(X, y, perceptron):
    # Calculate predictions
    y_pred = [perceptron.predict(x) for x in X]

    # Calculate misclassification error
    error = sum(y[i] != y_pred[i] for i in range(len(X))) / len(X)

    # Return misclassification error
    return error

def plot_data_and_boundary(X, y, w,title_1):
    plt.figure(figsize=(8,6))

    plt.scatter(X[:,0], X[:,1], c=y, cmap=plt.cm.coolwarm)
    plt.title(title_1)
    x_min, x_max = X[:, 0].min() - 0.5, X[:, 0].max() + 0.5
    y_min, y_max = X[:, 1].min() - 0.5, X[:, 1].max() + 0.5
    xx, yy = np.meshgrid(np.arange(x_min, x_max, 0.01),
                         np.arange(y_min, y_max, 0.01))
    c = np.c_[xx.ravel(), yy.ravel()]
    Z = np.dot(np.concatenate((np.ones((c.shape[0], 1)), c), axis=1), w)
    Z = Z.reshape(xx.shape)
    plt.contour(xx, yy, Z, colors=['k', 'k', 'k'], linewidths=0.75,linestyles=['--'


    plt.xlabel("X")
    plt.ylabel("Y")
    plt.xlim(xx.min(), xx.max())
    plt.ylim(yy.min(), yy.max())
    plt.xticks()
    plt.yticks()
    plt.show()
```

```
import os
import pandas as pd

# Set the path to the directory containing the data files
data_dir = '/content/drive/MyDrive/Colab Notebooks/Assignment_2/twoclassData'

# Load training data
train_files = sorted([f for f in os.listdir(data_dir) if f.startswith('set') and f.

train_data = []
for train_file in train_files:
    with open(os.path.join(data_dir, train_file), 'r') as f:
        data = pd.read_csv(f, sep=' ', header=None)
        data=data.dropna(axis='columns')
        data.iloc[:,-1] = data.iloc[:,-1].astype('int')
        #print(data.info())
        X = data.iloc[:, :-1].values
        y = data.iloc[:,-1].values
        train_data.append((X, y))
        #print(y)
        print(train_file)

# Load test data
test_file = 'set.test'

with open(os.path.join(data_dir, test_file), 'r') as f:
    data = pd.read_csv(f, sep=' ', header=None)
    data=data.dropna(axis='columns')
    data.iloc[:,-1] = data.iloc[:,-1].astype('int')
    #print(data.info())
    X_test = data.iloc[:,:-1].values
    y_test = data.iloc[:,-1].values
```

```
set1.train
set10.train
set2.train
set3.train
set4.train
set5.train
set6.train
set7.train
set8.train
set9.train
```

```
print(y_test)
```

```
[0 0 0 ... 1 1 1]
```

```
import matplotlib.pyplot as plt
total_error = []
for values in range(1, 101):
  error_rates = []
  print(f"num_epochs : {values}")
  print("\n\n")
  for i, (X_train, y_train) in enumerate(train_data):
    # Train perceptron
    perceptron, num_iterations = train_perceptron(X_train, y_train, num_epochs =

    # Test perceptron
    error = test_perceptron(X_test, y_test, perceptron)
    #print(f'Set {i+1}: Error={error:.4f}, Iterations={num_iterations//40}')
    error_rates.append(error)

    # Plot the training data and decision boundary for the current set
    title_1 = (f"Set {i+1} Training Data")
    plot_data_and_boundary(X_train, y_train, perceptron.weights,title_1)


    # Plot the test data and decision boundary for the current set
    title_1 = ("Set Test Data")
    plot_data_and_boundary(X_test, y_test, perceptron.weights,title_1)


  # Print the error rates for each set
  for i, error_rate in enumerate(error_rates):
    print(f"Set {i+1} error rate: {error_rate:.2f}")
  print(sum(error_rates))
  total_error.append(sum(error_rates))
  print("\n\n")
```

```
for i in range(len(total_error)):
  print(i,total_error[i])
```

```
41 0.252
42 0.10700000000000001
43 0.16350000000000003
```

```
44 0.136
45 0.1795
46 0.22749999999999998
47 0.154
48 0.195
49 0.18750000000000003
50 0.1265
51 0.1245
52 0.185
53 0.1895
54 0.197
55 0.17450000000000002
56 0.1265
57 0.256
58 0.2495
59 0.13299999999999998
60 0.2675
61 0.168
62 0.13
63 0.16150000000000003
64 0.2145
65 0.192
66 0.139
67 0.136
68 0.1285
69 0.2115
70 0.0855
71 0.1955
72 0.22700000000000004
73 0.172
74 0.26650000000000007
75 0.0985
76 0.1025
77 0.202
78 0.14
79 0.1715
80 0.161
81 0.20750000000000002
82 0.15700000000000003
83 0.2175
84 0.2355
85 0.13599999999999998
86 0.15800000000000003
87 0.241
88 0.1935
89 0.18550000000000003
90 0.147
91 0.242
92 0.11699999999999999
93 0.18350000000000002
```

```
94 0.1385
95 0.256
96 0.16850000000000004
97 0.206
98 0.189
99 0.15800000000000003
```

```
df = pd.DataFrame (total_error, columns = ['Total_Error'])
```

```
df.head()
```

	Total_Error
0	1.6940
1	1.6490
2	0.8570
3	0.5845
4	0.4440

```
df[['Total_Error']].idxmin()
```

```
Total_Error    36
dtype: int64
```

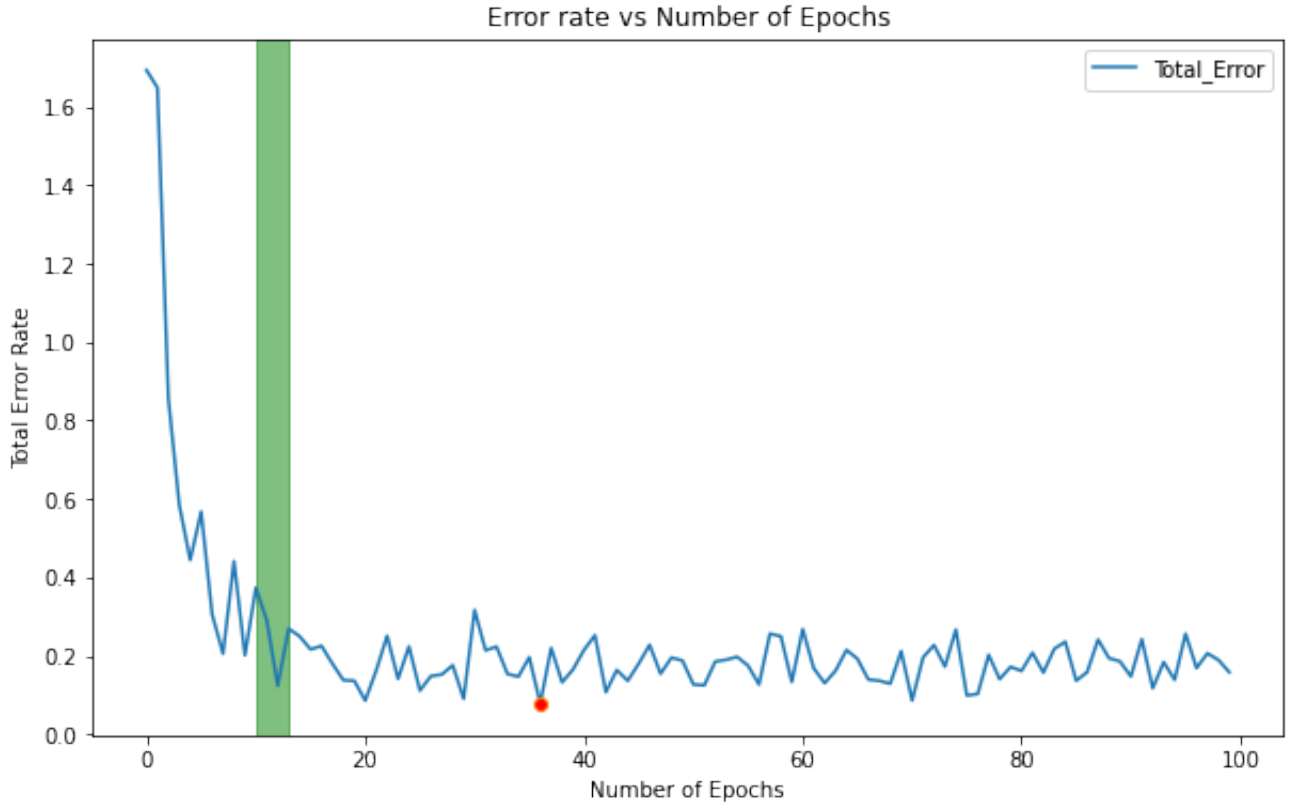
```
df['Total_Error'].min()
```

```
0.07750000000000001
```

```
ax = df.plot(y='Total_Error', use_index=True, title = "Error rate vs Number of Ep

ax.plot(df[['Total_Error']].idxmin(),df['Total_Error'].min(),'o',markerfacecolor=
ax.axvspan(10, 13, color='green', alpha=0.5)
```

```
<matplotlib.patches.Polygon at 0x7f0ecf8110d0>
```




```
fig = ax.get_figure()
fig.savefig("ER_vs_eps.png")
```

Colab paid products - Cancel contracts here

✓ 0s completed at 11:10 PM