CoxAssignment09

Task: This assignment takes you through exploring the Perceptron Algorithm that we used in class. This should give you a high-level overview of Neural Networks and the intuition behind them: https://towardsdatascience.com/exploring-the-perceptron-algorithm-using-python-c1d3af53a7c7

Dataset: Created by you

The idea for this assignment is not simply to copy and paste some code. There are interesting differences I want you to note in this code compared to what we have done. There are two major learning points here.

1. Scikit-Learn is a FANTASTIC resource and contains many of the things required to complete predictive or classification problems.
2. TowardsDataScience can be a great resource for learning and understanding the models. Neural Networks are very popular classification models.

To complete this assignment, type in all the code and discuss what each section of code is doing. As opposed to you thinking up the code to use, this assignment will challenge you to discuss the importance of step and note some of the differences between the code they have written and the code we have written.

In [1]:
```
import matplotlib.pyplot as plt
import numpy as np
plt.style.use('ggplot')
plt.rcParams['font.family'] = 'sans-serif'
plt.rcParams['font.serif'] = 'Ubuntu'
plt.rcParams['font.monospace'] = 'Ubuntu Mono'
plt.rcParams['font.size'] = 14
plt.rcParams['axes.labelsize'] = 12
plt.rcParams['axes.labelweight'] = 'bold'
plt.rcParams['axes.titlesize'] = 12
plt.rcParams['xtick.labelsize'] = 12
plt.rcParams['ytick.labelsize'] = 12
plt.rcParams['legend.fontsize'] = 12
plt.rcParams['figure.titlesize'] = 12
plt.rcParams['image.cmap'] = 'jet'
plt.rcParams['image.interpolation'] = 'none'
plt.rcParams['figure.figsize'] = (10, 10
                                 )
plt.rcParams['axes.grid']=True
plt.rcParams['lines.linewidth'] = 2
plt.rcParams['lines.markersize'] = 8
colors = ['xkcd:pale range', 'xkcd:sea blue', 'xkcd:pale red', 'xkcd:sage green', 'xkc
```

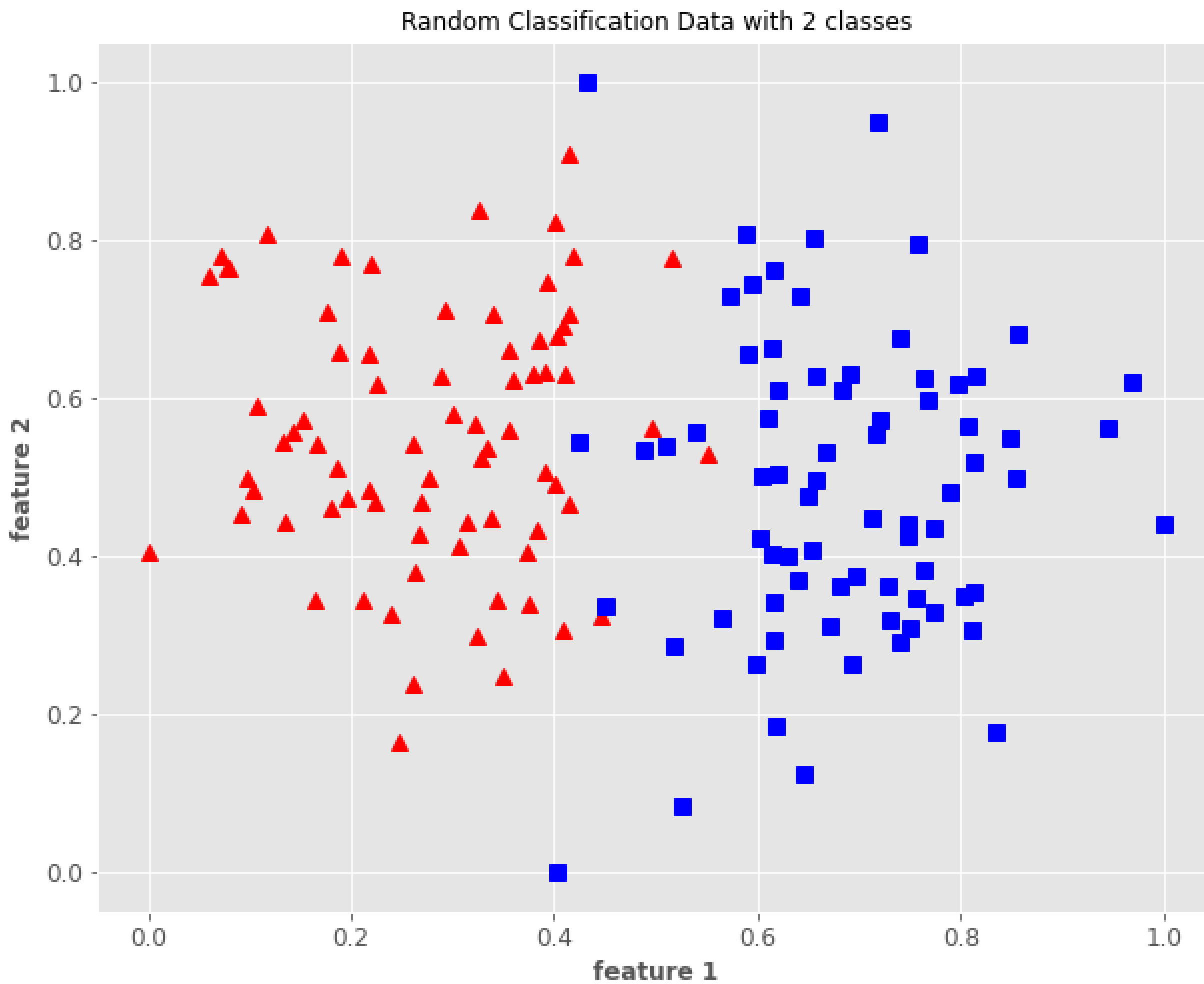
```
'xkcd:scarlet']
bbox_props = dict(boxstyle="round,pad=0.3", fc=colors[0], alpha=.5)
```

In [2]:
```
def step_func(z):
        return 1.0 if (z > 0) else 0.0
```

    - The decision function here is what the algorithm uses to help make
    a classification decision

In [18]:
```
from sklearn import datasets
from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import MinMaxScaler


X, y = datasets.make_blobs(n_samples=150,n_features=2,
                           centers=2,cluster_std=3.20)
y[y==0]=-1

#Plotting

min_max_scaler = MinMaxScaler()
X = min_max_scaler.fit_transform(X)

fig = plt.figure(figsize=(10,8))
plt.plot(X[:, 0][y == -1], X[:, 1][y == -1], 'r^')
plt.plot(X[:, 0][y == 1], X[:, 1][y == 1], 'bs')
plt.xlabel("feature 1")
plt.ylabel("feature 2")
plt.title('Random Classification Data with 2 classes')
```

Out[18]:
```
Text(0.5, 1.0, 'Random Classification Data with 2 classes')
```



- This lineraly separable dataset is called such since it is able to be split up via a single line that can seperate the two different sets

In [24]:
```
def perceptron(X, y, lr, epochs):

    # X --> Inputs.
    # y --> labels/target.
    # lr --> learning rate.
    # epochs --> Number of iterations.

    # m-> number of training examples
    # n-> number of features
    m, n = X.shape

    # Initializing parapeters(theta) to zeros.
    # +1 in n+1 for the bias term.
    theta = np.zeros((n+1,1))

    # Empty list to store how many examples were
    # misclassified at every iteration.
    n_miss_list = []
    loss_list = []
    # Training.
    for epoch in range(epochs):
```

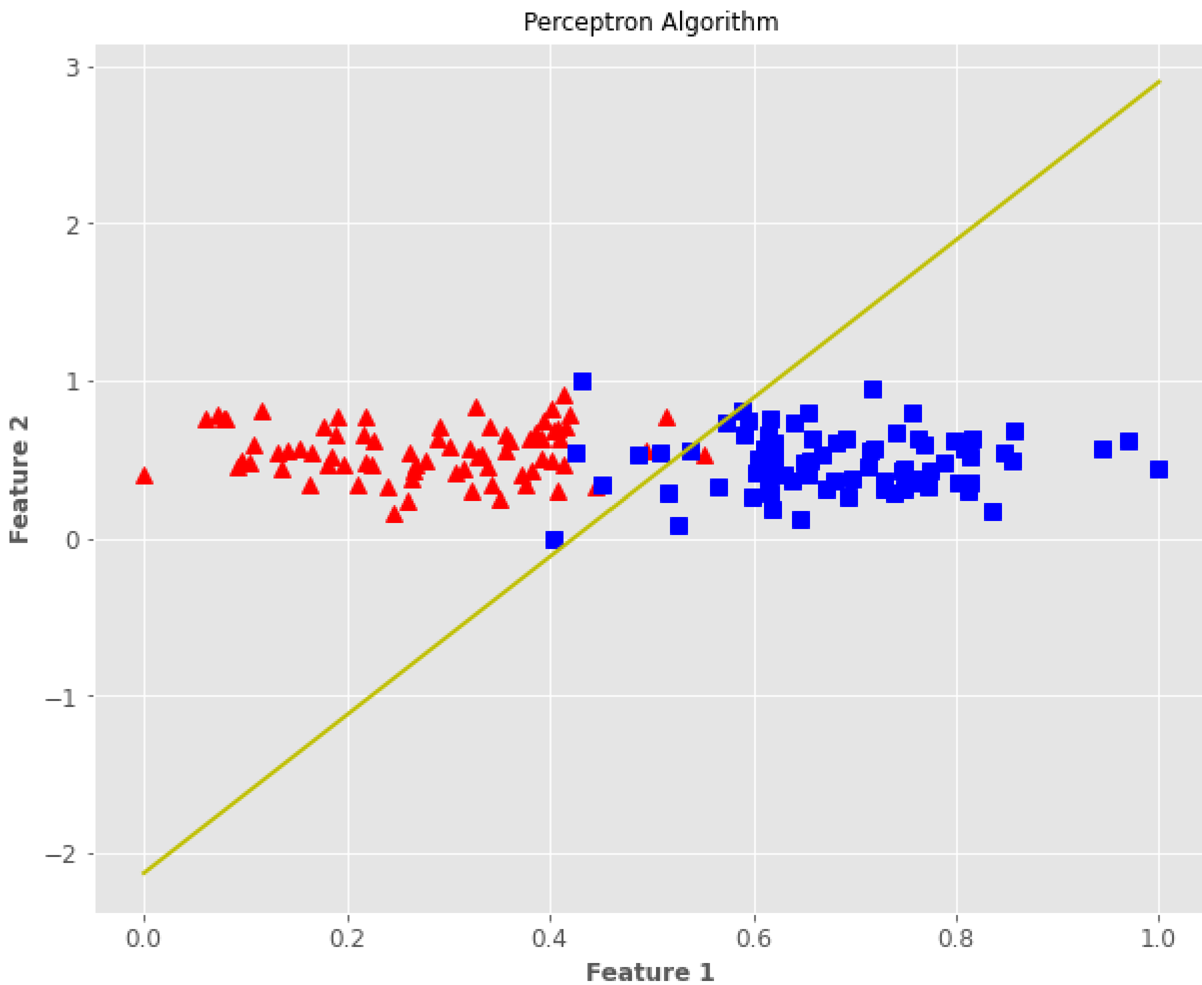
```
        # variable to store #misclassified.
        n_miss = 0

        # looping for every example.
        for idx, x_i in enumerate(X):

            # Insering 1 for bias, X0 = 1.
            x_i = np.insert(x_i, 0, 1).reshape(-1,1)

            # Calculating prediction/hypothesis.
            y_hat = step_func(np.dot(x_i.T, theta))
            if y_hat==0:
              y_hat = -1
            # Updating if the example is misclassified.
            if (np.squeeze(y_hat) - y[idx]) != 0:
                theta += lr*((y[idx] - y_hat)*x_i)

                # Incrementing by 1.
                n_miss += 1
        #Defining the loss function
        x1 = X[:,0]
        x2 = X[:,1]
        theta_array = theta
        loss_value = (theta_array[1]*x1+theta_array[2]*x2+theta_array[0])*y
        loss_value = loss_value.sum()/len(x1)
        loss_list.append(loss_value)
        # Appending number of misclassified examples
        # at every iteration.
        n_miss_list.append(n_miss)

    return theta, n_miss_list,loss_list
```

```
- This takes the inputs and the target variable and combines it with
the learning rate to enhance the training set via epochs. Then it
takes the loss function to shows the cost of an event. Idealy you
want to minimize the loss function as much as possible.
```

In [25]:
```
def plot_decision_boundary(X, theta):

    # X --> Inputs
    # theta --> parameters

    # The Line is y=mx+c
    # So, Equate mx+c = theta0.X0 + theta1.X1 + theta2.X2
    # Solving we find m and c
    x1 = [min(X[:,0]), max(X[:,0])]
    m = -theta[1]/theta[2]
    c = -theta[0]/theta[2]
    x2 = m*x1 + c

    # Plotting
    fig = plt.figure(figsize=(10,8))
    plt.plot(X[:, 0][y==-1], X[:, 1][y==-1], "r^")
    plt.plot(X[:, 0][y==1], X[:, 1][y==1], "bs")
    plt.xlabel("Feature 1")
    plt.ylabel("Feature 2")
    plt.title('Perceptron Algorithm')
    plt.plot(x1, x2, 'y-')
```

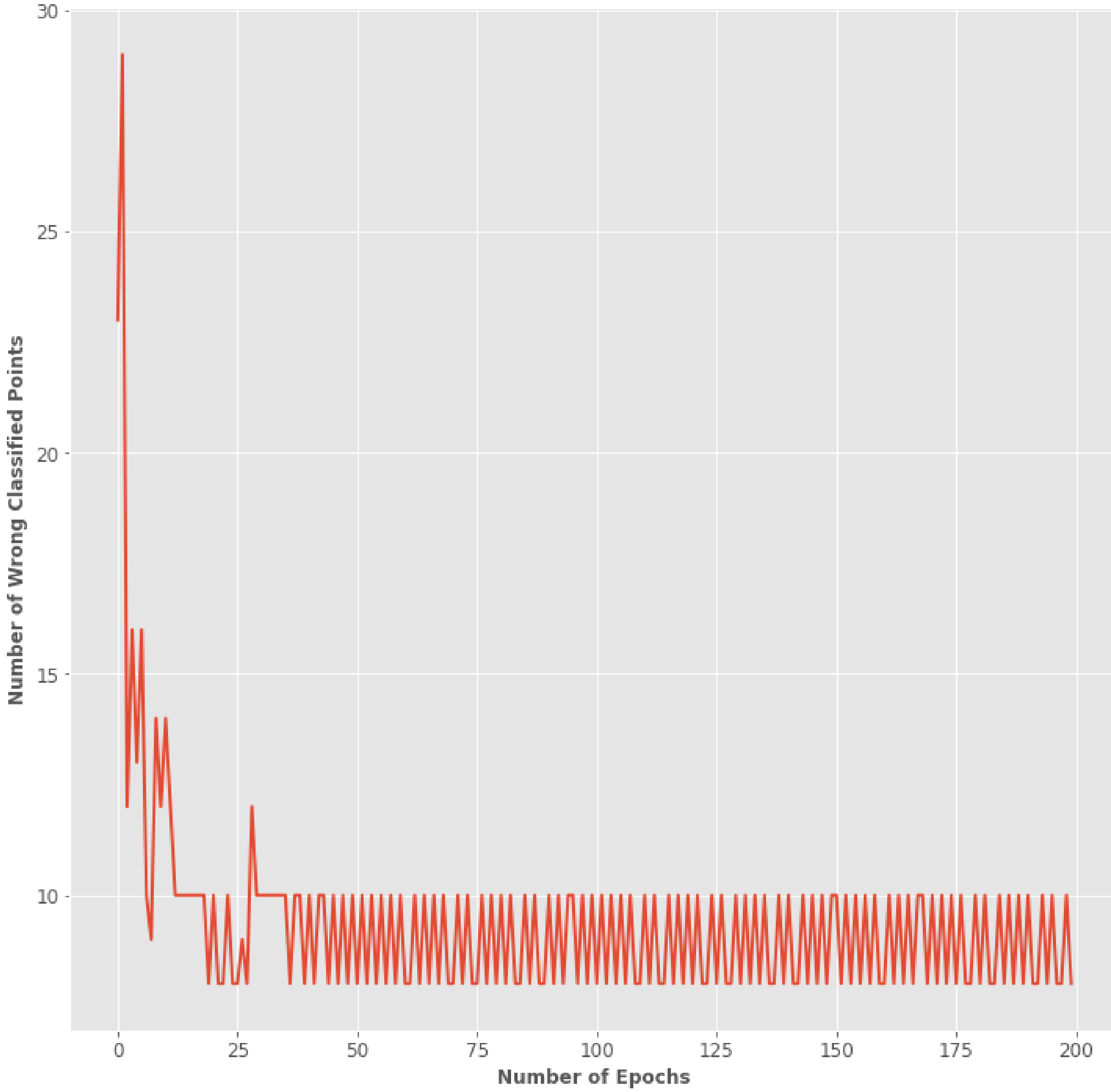
- The decision boundary is the part of the problem where the output is vague

In [26]:
```
learning_rate , epoch = 0.005,200
theta, miss_l,loss_list= perceptron(X, y, learning_rate, epoch)
plot_decision_boundary(X, theta)
```



- From this you can see that the majority of the data is well classified. There are a few blue squares that cross over and one red triangle but other than that the vast majority of them are classified properly.

In [27]:
```
def plot_training(miss_l):
  plt.figure(figsize=(12,12))
  list_array = np.arange(0,len(miss_l),1)
  plt.xlabel('Number of Epochs')
  plt.ylabel('Number of Wrong Classified Points')
  plt.plot(list_array,miss_l)
plot_training(miss_l)
```

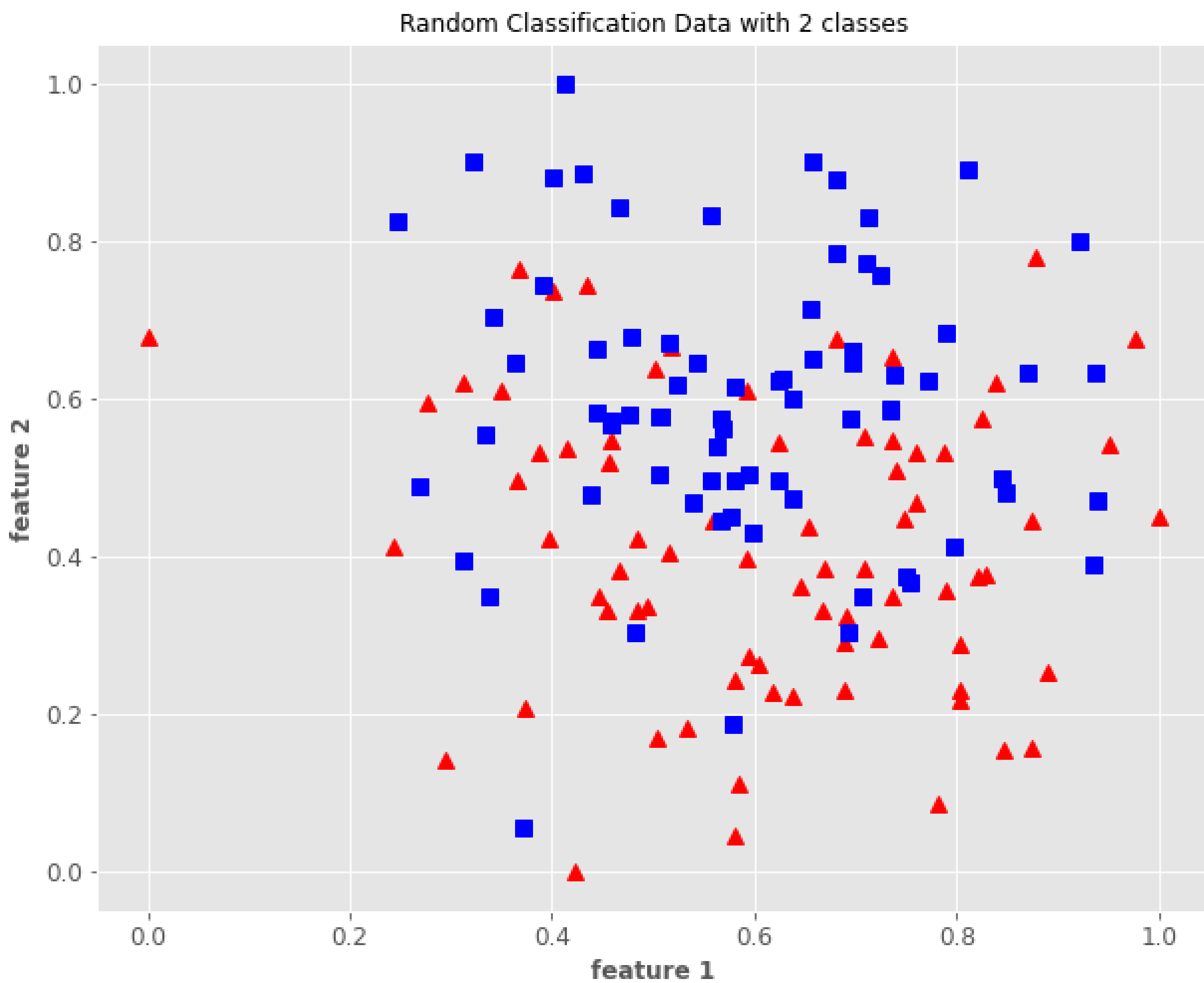


- This shows that the data is not perfectly classified as you can see with the up and down markers as the number of epochs increase. Ideally it would be a straight line at 0 for number of wrong classified points.

In [28]:
```
from sklearn import datasets
X, y = datasets.make_blobs(n_samples=150,n_features=2,
                           centers=2,cluster_std=3.20)
y[y==0]=-1

#Plotting

min_max_scaler = MinMaxScaler()
X = min_max_scaler.fit_transform(X)

fig = plt.figure(figsize=(10,8))
plt.plot(X[:, 0][y == -1], X[:, 1][y == -1], 'r^')
plt.plot(X[:, 0][y == 1], X[:, 1][y == 1], 'bs')
plt.xlabel("feature 1")
```

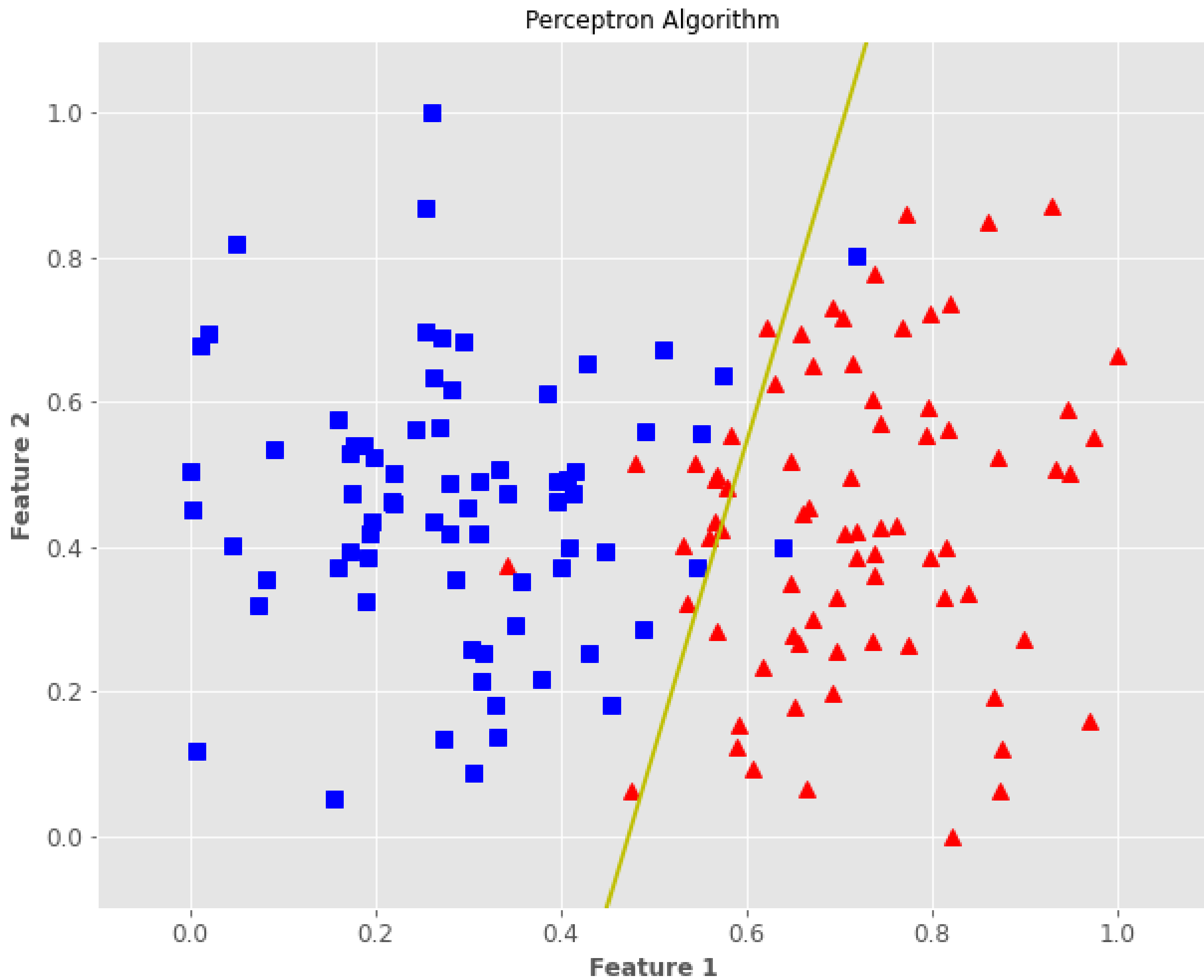
```
plt.ylabel("feature 2")
plt.title('Random Classification Data with 2 classes')
```

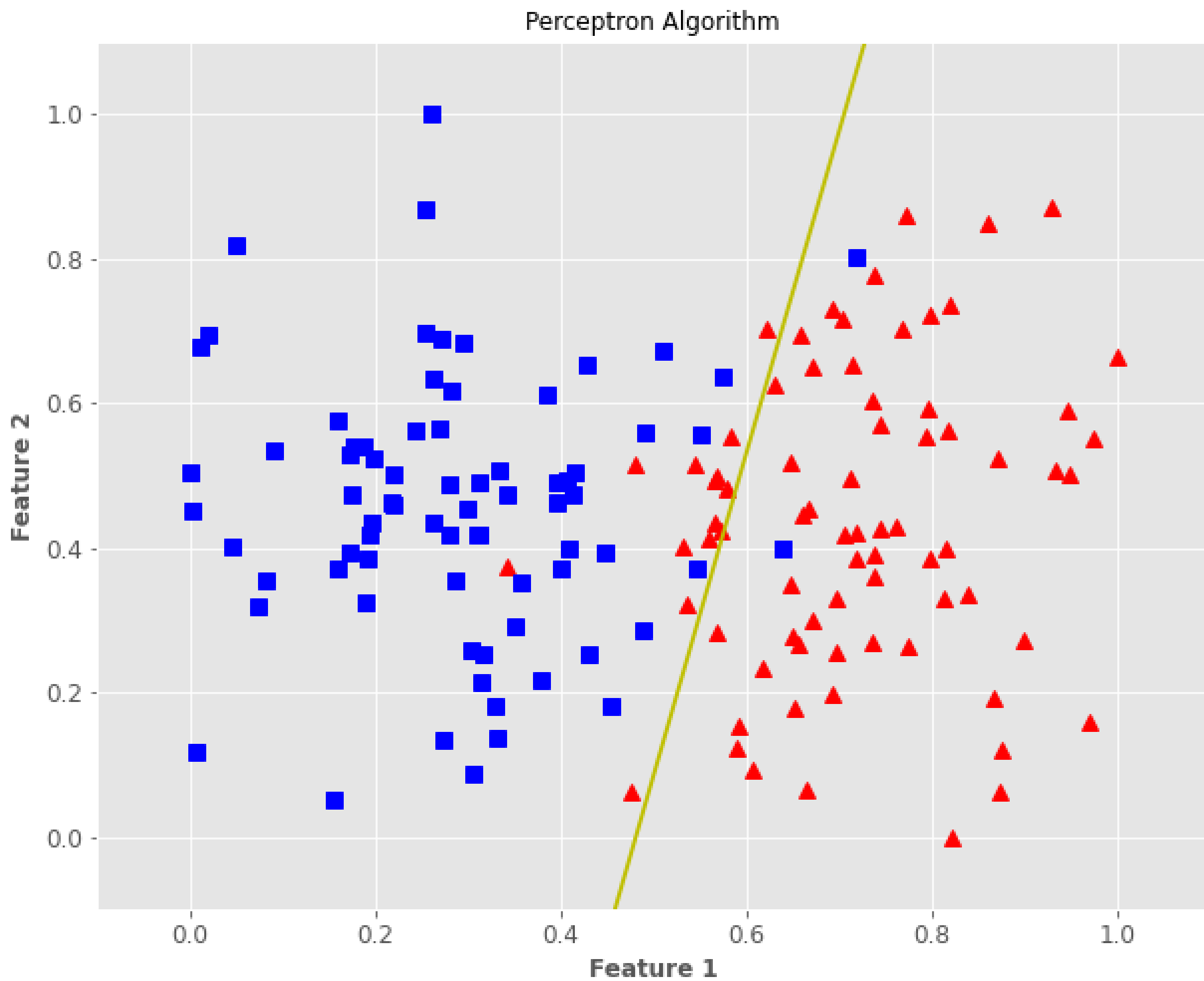
Out[28]: Text(0.5, 1.0, 'Random Classification Data with 2 classes')



In [29]:
```
def plot_decision_boundary(X, theta):

    # X --> Inputs
    # theta --> parameters

    # The Line is y=mx+c
    # So, Equate mx+c = theta0.X0 + theta1.X1 + theta2.X2
    # Solving we find m and c
    x1 = [min(X[:,0]), max(X[:,0])]
    m = -theta[1]/theta[2]
    c = -theta[0]/theta[2]
    x2 = m*x1 + c

    # Plotting
    fig = plt.figure(figsize=(10,8))
    plt.plot(X[:, 0][y==-1], X[:, 1][y==-1], "r^")
    plt.plot(X[:, 0][y==1], X[:, 1][y==1], "bs")
    plt.xlabel("Feature 1")
    plt.ylabel("Feature 2")
    plt.title('Perceptron Algorithm')
    plt.plot(x1, x2, 'y-')
    plt.xlim(-0.1,1.1)
    plt.ylim(-0.1,1.1)
```

In [10]:
```
theta, miss_l,loss = perceptron(X, y, 0.2, 10)

plot_decision_boundary(X, theta)

theta, miss_l,loss = perceptron(X, y, 1, 20)

plot_decision_boundary(X, theta)
```



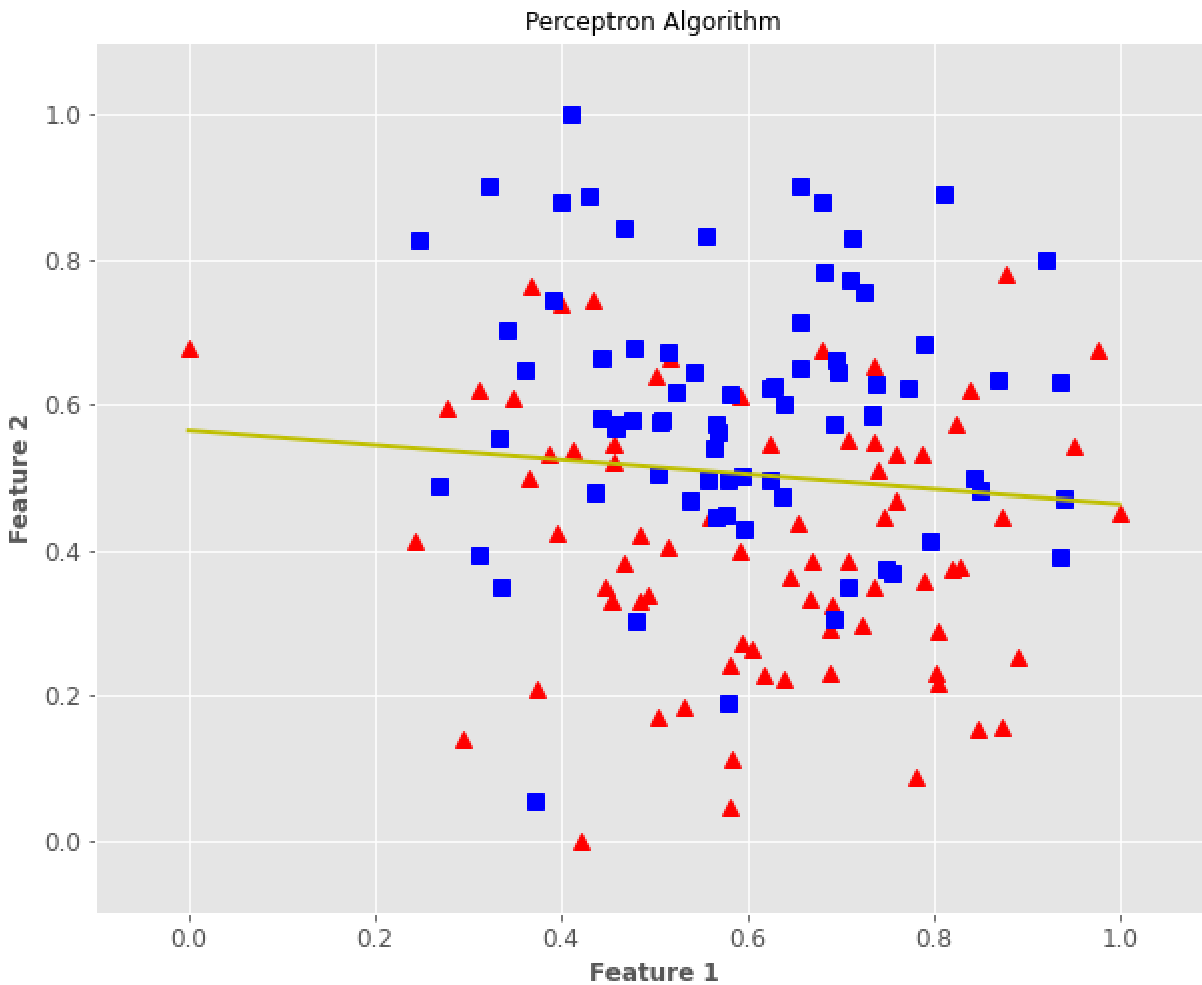


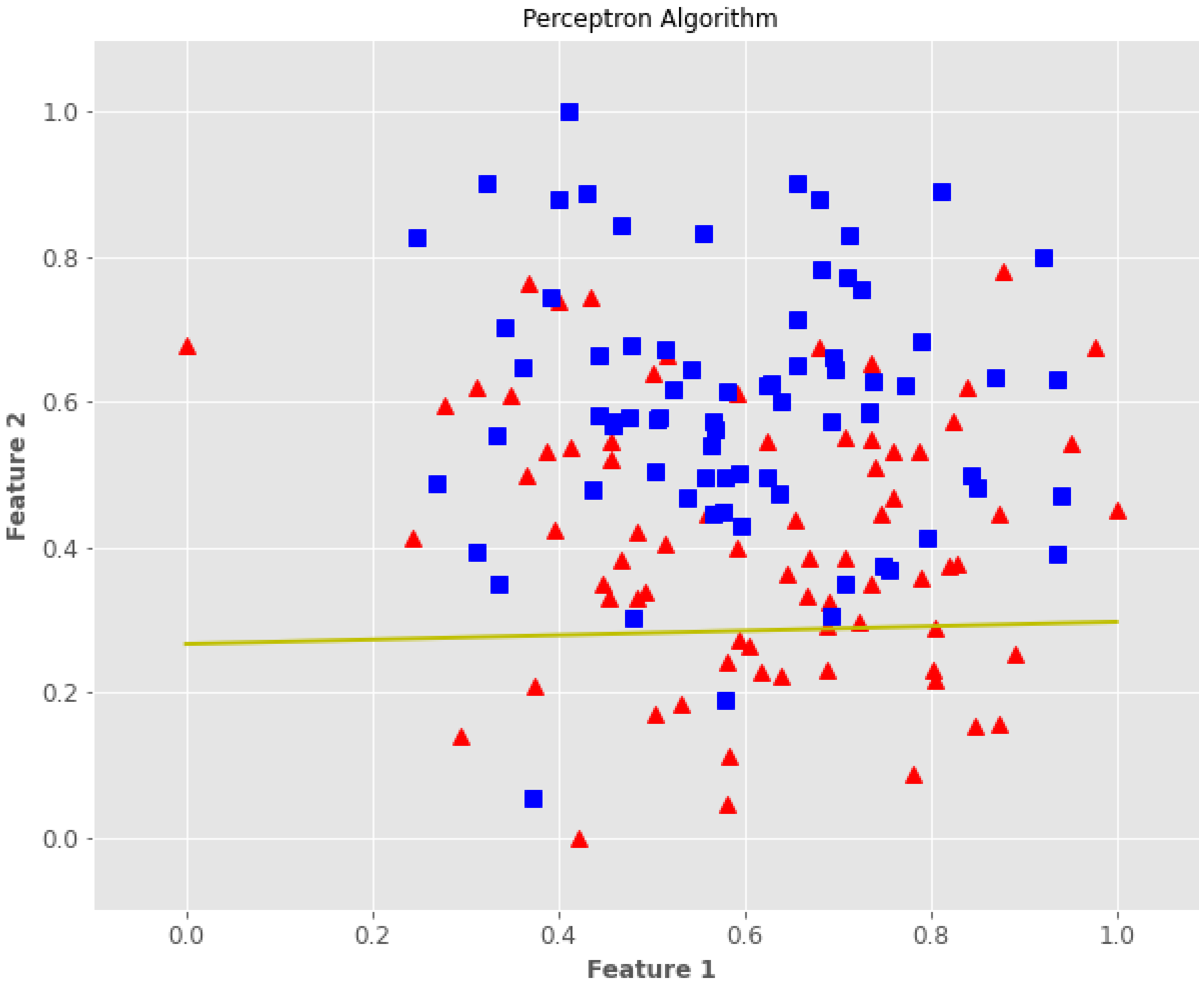
- This time we are running the same code as above but with much harder to classify data. We will be using hyperparamter tuning to get the best possible version we can.

In [30]:
```
def plot_decision_boundary(X, theta):

    # X --> Inputs
    # theta --> parameters

    # The Line is y=mx+c
    # So, Equate mx+c = theta0.X0 + theta1.X1 + theta2.X2
    # Solving we find m and c
    x1 = [min(X[:,0]), max(X[:,0])]
    m = -theta[1]/theta[2]
    c = -theta[0]/theta[2]
    x2 = m*x1 + c

    # Plotting
    fig = plt.figure(figsize=(10,8))
    plt.plot(X[:, 0][y==-1], X[:, 1][y==-1], "r^")
    plt.plot(X[:, 0][y==1], X[:, 1][y==1], "bs")
    plt.xlabel("Feature 1")
    plt.ylabel("Feature 2")
    plt.title('Perceptron Algorithm')
    plt.plot(x1, x2, 'y-')
```

```
    plt.xlim(-0.1,1.1)
    plt.ylim(-0.1,1.1)
```

In [31]:
```
theta, miss_l,loss = perceptron(X, y, 0.2, 10)

plot_decision_boundary(X, theta)

theta, miss_l,loss = perceptron(X, y, 1, 20)

plot_decision_boundary(X, theta)
```





- Now that the dataset it not as perfect as it could be for being able to classify. The following code will be used to help make the most out of what we have.

In [32]:
```
from sklearn.linear_model import Perceptron
num_of_epochs = [10,100,500,1000]
etas = np.linspace(1e-5,1,100)
scores = []
for e in etas:
  for num in num_of_epochs:
    clf = Perceptron(eta0=e,max_iter=num)
    clf.fit(X, y)
    scores.append({'Num':num,'Eta':e.round(5),'Score':clf.score(X, y)})
```

```
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
```

```
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
```

```
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
```

```
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
```

```
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
```

```
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
```

```
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\jcjcb\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.
py:696: ConvergenceWarning: Maximum number of iteration reached before convergence. C
onsider increasing max_iter to improve the fit.
  warnings.warn(
```

In [33]:

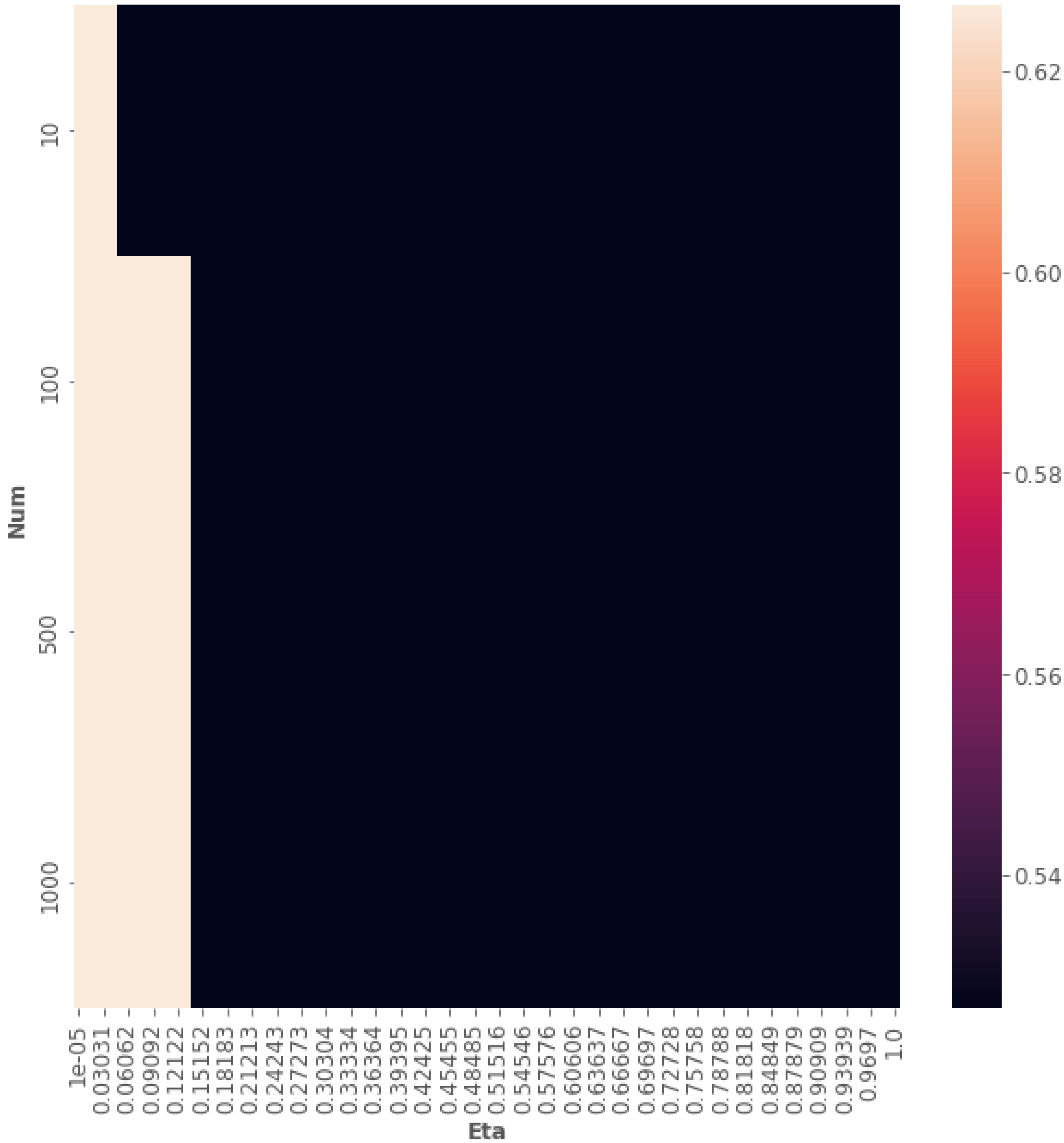
```
import pandas as pd
import seaborn as sns
scores=pd.DataFrame(scores)
pivot = scores.pivot('Num','Eta','Score')
```

In [34]:

```
sns.heatmap(data=pivot)
```

Out[34]:

```
<AxesSubplot:xlabel='Eta', ylabel='Num'>
```



In [35]:
```
scores[scores.Score==scores.Score.max()]
```

Out[35]:

	Num	Eta	Score
0	10	0.00001	0.626667
1	100	0.00001	0.626667
2	500	0.00001	0.626667
3	1000	0.00001	0.626667
4	10	0.01011	0.626667
5	100	0.01011	0.626667
6	500	0.01011	0.626667
7	1000	0.01011	0.626667
8	10	0.02021	0.626667
9	100	0.02021	0.626667
10	500	0.02021	0.626667
11	1000	0.02021	0.626667
12	10	0.03031	0.626667
13	100	0.03031	0.626667
14	500	0.03031	0.626667
15	1000	0.03031	0.626667
16	10	0.04041	0.626667
17	100	0.04041	0.626667
18	500	0.04041	0.626667
19	1000	0.04041	0.626667
21	100	0.05051	0.626667
22	500	0.05051	0.626667
23	1000	0.05051	0.626667
25	100	0.06062	0.626667
26	500	0.06062	0.626667
27	1000	0.06062	0.626667
29	100	0.07072	0.626667
30	500	0.07072	0.626667
31	1000	0.07072	0.626667
33	100	0.08082	0.626667
34	500	0.08082	0.626667
35	1000	0.08082	0.626667
37	100	0.09092	0.626667

	Num	Eta	Score
38	500	0.09092	0.626667
39	1000	0.09092	0.626667
41	100	0.10102	0.626667
42	500	0.10102	0.626667
43	1000	0.10102	0.626667
45	100	0.11112	0.626667
46	500	0.11112	0.626667
47	1000	0.11112	0.626667
49	100	0.12122	0.626667
50	500	0.12122	0.626667
51	1000	0.12122	0.626667
53	100	0.13132	0.626667
54	500	0.13132	0.626667
55	1000	0.13132	0.626667

```
-These are the optimal number of epochs and learning rate for our
dataset. Given how low the score is, it it probaly fair to assume
that this data is not the best fit for what we are trying to do. But
these are the optimal number of epochs and learning rate if we decide
to move forward. Based on the hyperparameter tuning aspect we could
increase the performance of our algorithm.
```

In [ ]: