

Weather probability in Australia

Abstract:

The main idea of this project is to be able to predict if its going to rain the next day in Australia. I use a 10 years Data Set of numerous Australian weather stations.

The data:

The data set used for this project was obtained in Kaggle, this is a real data set taken from 10 years of measurements made by numerous Australian weather stations.

Link: <https://www.kaggle.com/jsphyg/weather-dataset-rattle-package>

The data set has 142k instances, but the program doesn't really need that much, so I only used 4000 instances. And I only use 13 weights to make the prediction work.

Arranging and cleaning the data:

I use a simple split function, so I get half of the data set that I have and use it for training and then the other half for testing. Then we remove the titles row, and all the rows that are incomplete. With this we are sure that we are only going to use complete data.

Algorithm:

I use the logistic regression algorithm to get the hypothesis and the gradients needed to the logistic part of the program.

```
def hy(weights, x):  
  
    """ This function makes the hypothesis for the Logistic regressios  
  
    weights: are the parameters that multiplies their respective x  
  
    x: are the values from one sample
```

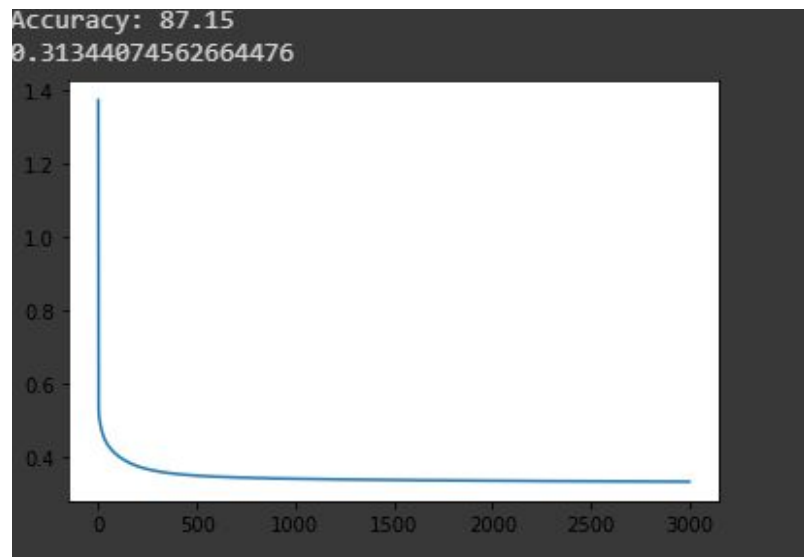
```
"""  
  
hyp = 0  
  
for i in range(len(weights)):  
  
    hyp+=(weights[i]*x[i])  
  
hyp = hyp * (-1)  
  
hyp = 1/(1+ math.exp (hyp))  
  
return hyp  
  
def gD(weights,x,y,a):  
  
    """ Gradient descent function """  
  
    temp=list(weights)  
  
    for j in range(len(weights)):  
  
        acum=0  
  
        for i in range(len(x)):  
  
            err=hy(weights,x[i])-y[i]  
  
            acum+=err*x[i][j]  
  
        temp[j]=weights[j]-a*(1/len(x))*acum  
  
    return temp
```

Running:

When running the program prints the error of that epoch, we use logarithm error to get the difference between the hypothesis and the real output. After a certain number of epochs, or if the error is lower than .20, the program stops. When it stops you get a plot of the error, and a final program gives you the accuracy of the program, this part is tested with data that the program has never seen. If the hyp is higher than .5 then it's considered as a 1 if it's lower than a .5 it's considered as a 0.

```
logError 0.018556 hyp 0.018385 activation 0 y 0.000000
logError 0.012581 hyp 0.012502 activation 0 y 0.000000
logError 0.054735 hyp 0.053264 activation 0 y 0.000000
logError 0.188629 hyp 0.828094 activation 1 y 1.000000
logError 0.373635 hyp 0.688228 activation 1 y 1.000000
logError 0.213932 hyp 0.192596 activation 0 y 0.000000
logError 0.019401 hyp 0.980786 activation 1 y 1.000000
logError 0.096198 hyp 0.908284 activation 1 y 1.000000
logError 3.937942 hyp 0.019488 activation 0 y 1.000000
logError 0.031365 hyp 0.030878 activation 0 y 0.000000
logError 3.070665 hyp 0.046390 activation 0 y 1.000000
logError 0.014490 hyp 0.014386 activation 0 y 0.000000
logError 0.029257 hyp 0.028833 activation 0 y 0.000000
logError 0.058197 hyp 0.056536 activation 0 y 0.000000
logError 0.097065 hyp 0.092503 activation 0 y 0.000000
logError 0.007023 hyp 0.006998 activation 0 y 0.000000
logError 0.054198 hyp 0.052755 activation 0 y 0.000000
logError 0.037821 hyp 0.037115 activation 0 y 0.000000
logError 0.033120 hyp 0.032577 activation 0 y 0.000000
```

The image shows what the test program print



This shows the accuracy and the error graph