Artificial Intelligence

Date of submission 20-12-2019.

*Session 2017/Section B/Semester Fall 2019*

Binary Classifier Using Perceptron Model

# Hammad Ali(2017-CS-104)

Corresponding author: Haseeb Virk (haseebvirk94@gmail.com ).

**ABSTRACT**

Pecptron is a very basic machine learning algorithm. In this project perceptron is used for simple binary classification. Threshold was used as the activation function for perceptron in this project.

**INTRODUCTION**

This is a console based project. The program takes a file containing training data as input. It learns from training data and saves the weights learnt from the each example in the given dataset. After the learning process is completed, the final weights can be used for predictions. Program takes a file as input for prediction. It predicts the output for the given examples in dataset and compares it with actual answer to find accuracy. The dataset for learning and prediction need to be in a specific format. A preprocessing functionality is also available to modify the dataset if its not same as given format

**DATA SET DETAILS**

The program needs dataset to be in following format to work:

|  |  |  |  |
| --- | --- | --- | --- |
| Column1 | Column2 | ………….. | Output |
| Numeric Value | Numeric Value | ………….. | 0 or 1 |
| Numeric Value | Numeric Value | ………….. | 0 or 1 |

Dataset must be in a csv file. Two example datasets (‘iris.csv’ and ‘andgate.csv’) are given along with the project as an example.

**DATA PREPROCESSING DETAILS**

Following options are given for pre processing data:

1. Change all values in a column to 0 and 1. For example if a column has genders in it and male is passed as main class name than if male is present anywhere inside that column it will change to 1 and all values will change to 0.
2. Remove a column from dataset.
3. Remove a row from dataset.

Example:

Iris dataset was taken from kaggle. In the original dataset “id” column was redundant so I removed it and the last row did not contain 0s and 1s. It contained names of the species of flowers. I changed it to make our datset a binary classification example of iris-setosa flower. So now the last contain 1 if it was iris-setosa else it contain 0.

**CODE OF PREPROCESSING**

import pandas

import sys

data\_csv=pandas.DataFrame()

out\_data=pandas.DataFrame()

data=[]

classifier=0

className=""

arguments=sys.argv

def classify():

    for i in range(0,len(data)):

        if data[i][classifier]==className:

            data[i][classifier]=1

        else:

            data[i][classifier]=0

        out\_data.loc[len(out\_data)]=data[i]

if arguments[1]=='--classify':

    if len(arguments)==7:

        data\_csv=pandas.read\_csv(arguments[2])

        data=data\_csv.values.tolist()

        out\_data=pandas.DataFrame(columns=data\_csv.columns)

        classifier=int(arguments[4])

        className=arguments[6]

        classify()

        out\_data.to\_csv("preprocessed.csv", index=False)

if arguments[1]=='--remove':

    data\_csv=pandas.read\_csv(arguments[2])

    if arguments[3]=="--column":

        out\_data=data\_csv.drop(data\_csv.columns[int(arguments[4])] , axis=1)

    elif arguments[3]=="--row":

        out\_data=data\_csv.drop(int(arguments[4]))

    out\_data.to\_csv("preprocessed.csv", index=False)

**METHOD**

The Perceptron algorithm:

1. Read Data from the given file
2. Initialize weights for each feature and threshold randomly.
3. Do following steps for each example given in file:
   1. Predict an output.
   2. Compare predicted output with actual output to find error.
   3. Update weights and threshold if error exists.
4. Perform step 3 again and again till there is no error in all examples given in the file.

**Code of Program:.**

# Importing necessory modules

import pandas

import random

import sys

# Declaring global variables

learning\_rate=0.2

data\_csv=pandas.DataFrame()

data=[]

weights=[]

prediction=[]

activation=[]

arguments=sys.argv

col\_names=[]

# Function to fix threshold

def fixThreshold(error):

    newBiase=weights[0]+learning\_rate\*(error)

    if newBiase>learning\_rate and newBiase<1 :

        weights[0]=newBiase

# Function to predict output for an amaple in dataset

def predict(row):

    activation=0

    for i in range(1,len(weights)):

        activation+=weights[i]\*row[i-1]

    activation-=weights[0]

    return 1 if activation>=0 else 0

# Function to update the weights according to the error

def errorSolving(error,row):

    for i in range (1,len(weights)):

        weights[i]=weights[i]+learning\_rate\*(error)\*row[i-1]

# Function to initialize weights and threshold randomly

def initialize():

    weights.append(random.uniform(learning\_rate,1))

    for j in range(0,len(data\_csv.columns)-1):

        ran=random.uniform(0,1)

        weights.append(ran)

# Main entry point function to start learning

def learn():

    flag=True

    while flag==True:

        flag=False

        for i in range(0,len(data)):

            value=predict(data[i])

            actualOutput=data[i][len(data\_csv.columns)-1]

            error=actualOutput-value

            errorSolving(error,data[i])

            fixThreshold(error)

            save()

            if error!=0.0:

                flag=True

    print("Done")

# Function to find Accuracy

def Accuracy():

    data\_csv=pandas.read\_csv('testpredict.csv')

    data=data\_csv.values.tolist()

    actualIndex=len(data\_csv.columns)-2

    predictedIndex=len(data\_csv.columns)-1

    correct=0

    Truepositive=0

    for i in range(0,len(data)):

        if data[i][actualIndex]==data[i][predictedIndex]:

            correct=correct+1

    Accuracy=correct/len(data)\*100

    print("Accuracy is: "+str(Accuracy)+"%")

# Function to save weights calculated in each episode in a file

def save():

    values=[]

    for i in range(0,len(weights)):

        values.append(weights[i])

    mdf.loc[len(mdf)]=values

    print(values)

    if len(arguments)<4:

        mdf.to\_csv("data.csv", index=False)

    else:

        mdf.to\_csv(arguments[4]+".csv", index=False)

if arguments[1]=="--learn":

    data\_csv=pandas.read\_csv(arguments[2])

    data=data\_csv.values.tolist()

    initialize()

    for i in range(0,len(weights)):

        col\_names.append('w'+str(i))

    mdf=pandas.DataFrame(columns=col\_names)

    values=[]

    for i in range(0,len(weights)):

        values.append(weights[i])

    mdf.loc[len(mdf)]=values

    learn()

if arguments[1]=="--test":

    data\_csv=pandas.read\_csv(arguments[2])

    data=data\_csv.values.tolist()

    if len(arguments)<4:

        learningFile\_csv=pandas.read\_csv("data.csv")

        learningFile=learningFile\_csv.values.tolist()

    else:

        learningFile\_csv=pandas.read\_csv(arguments[4])

        learningFile=learningFile\_csv.values.tolist()

    prdictedValue=0

    for i in range(1,len(data\_csv.columns)):

        col\_names.append('x'+str(i))

    col\_names.append('actualAnswer')

    col\_names.append('Predicted')

    mdf=pandas.DataFrame(columns=col\_names)

    for i in range(0,len(data)):

        values=[]

        activation=0

        for j in range(0,len(data\_csv.columns)-1):

            values.append(data[i][j])

            activation+=learningFile[len(learningFile)-1][j+1]\*data[i][j]

        activation=activation-learningFile[len(learningFile)-1][0]

        if activation>=0:

            predictedValue=1

        else:

            predictedValue=0

        k=len(data\_csv.columns)-1

        values.append(data[i][k])

        values.append(predictedValue)

        mdf.loc[len(mdf)]=values

        mdf.to\_csv("testpredict.csv", index=False)

    print("Done")

    Accuracy()

# EXPERIMENT AND RESULTS

# After testing the final results are stored in file and accuracy is calculated. Andgate orgate and iris datasets were used to test this project and the accuracy was always 100%.

# INSTRUCTION FOR PREPROCESSING DATA:

1. Download the project reposity from <https://github.com/hammad11ali/perceptron>
2. Open the project directory in command prompt.
3. Delete any row in which some value is missing. To delete a row write follwong command:

python preprocess.py –-remove *filename*.csv --row *column\_index*

This will delete the column at specified column\_index in the specified file.

1. Delete any rredundent column in data. To delete a column write following command

python preprocess.py –-remove *filename*.csv --row *row\_index*

This will delete the row at specified row\_index in the specified file.

1. Make the last column a input to 1s and 0s. To change a column to 0s and 1s write following command:

python preprocess.py –-classify *filename*.csv --column *columnindex* --name *name*

Where name is the name of the value which you want to change to 1. All other values will be changed to zero

# INSTRUCTION TO EXECUTE THE PROGRAM

1. Download the project reposity from <https://github.com/hammad11ali/perceptron>
2. Open the project directory in command prompt.
3. Preprocess data using the steps given above if necessary.
4. Start leaning process using following command:

python perceptron.py –-learn *filename*.csv

1. Test using following command:

python percept2.py –-test *filename*.csv

Note: Format for test and learn file is same

# REFERENCES

Dataset was taken from:

<https://www.kaggle.com/uciml/iris>