# Topic: Neural Network

**Name: Shubham Kumar Sharma**

**Email: shubhrock777@gmail.com**

**Topic: Neural Network**

**Problem Statement:-**

Abstract—

The choice of Activation Functions (AF) has proven to be an important factor that affects the performance of an Artificial Neural Network (ANN). Use a 1-hidden layer neural network model that adapts to the most suitable activation function according to the data-set. The ANN model can learn for itself the best AF to use by exploiting a flexible functional form, k0 + k1 ∗ x with parameters k0, k1 being learned from multiple runs. You can use this code-base for implementation guidelines and help. https://github.com/sahamath/MultiLayerPerceptron I.

BACKGROUND

Selection of the best performing AF for classification task is essentially a naive (or brute-force) procedure wherein, a popularly used AF is picked and used in the network for approximating the optimal function. If this function fails, the process is repeated with a different AF, till the network learns to approximate the ideal function. It is interesting to inquire and inspect whether there exists a possibility of building a framework which uses the inherent clues and insights from data and bring about the most suitable AF. The possibilities of such an approach could not only save significant time and effort for tuning the model, but will also open up new ways for discovering essential features of not-so-popular AFs.

**Business Objective- build a ANN model that predict species of iris**.

**Python code details :**

Data Frame name is model\_df . It has 150 entries and 5 features.

**Work on each feature of the dataset to create a data dictionary as displayed in the below image:**

Then I have create a data frame that’s contain details of each columns ,like- description ,data types ,and save the details named as model\_df\_details. All of them are important .

**Data Pre-processing**

**Data Cleaning and Data Mining.**

Now we check info and describe for df .Check for data types ,unique value and variance . we have changed Columns name to 'rd\_spend', 'Administration', 'm\_spend', 'State', 'Profit'.

Then we check for unique value for categorical columns

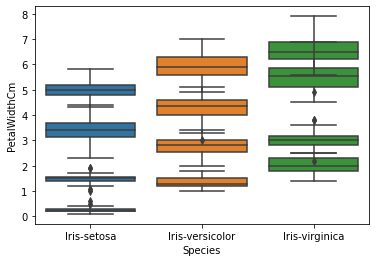
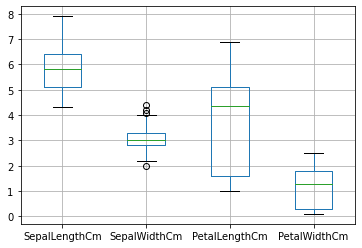
:-

#species have 3 factor

Dataframe has no missing values in columns .

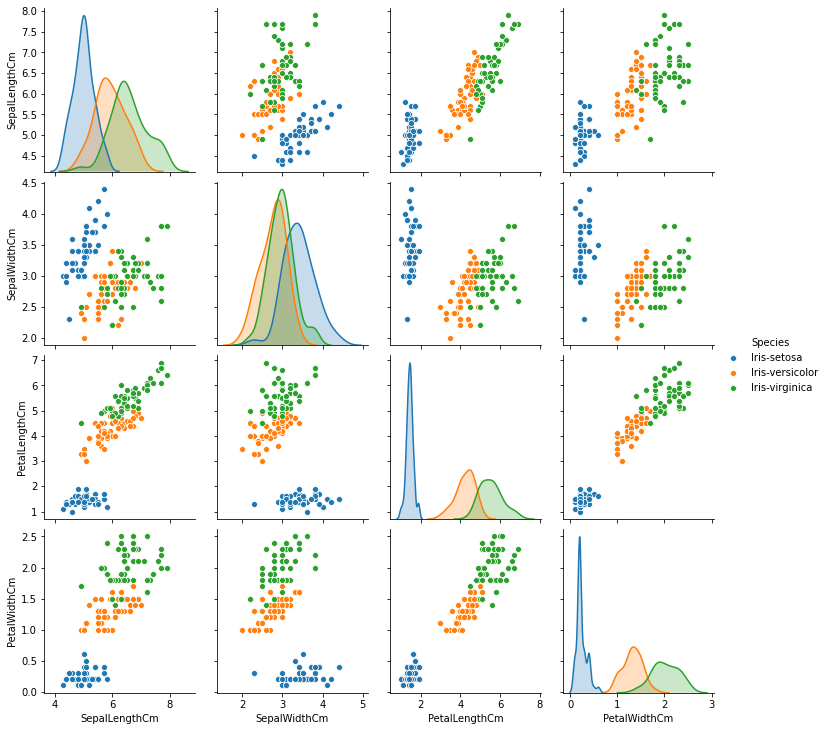
We have done EDA for each columns and save the details as EDA. covariance for data set save as covariance . historgam and scatter plot for each column all data are normally distributed as well as we check for boxplot .there is no outliers present except Profit.

Boxplot:-



In sepalwidth we can see some outlier but that is false positive

Histogram and Scatter plot:- for continuous data



**-**There is high correlation b/w petal\_length\_cm and petal\_width\_cm so we can also go for PCA.

If we take a look at our Dataset we can clearly see that species is a String type variable and like we have discussed,We cannot feed String type variables into our Machine Learning model as it can only work with numbers.

Using seaborn (pairplot) in python we can check for distribution and correlation between each other. According scatter plot strong positive correlation between petal\_length\_cm and petal\_width\_cm and also some relation between others.

**Correlation save as corr**

**there is high correlation b/w petal\_length\_cm and petal\_width\_cm-0.96**

**Model Building:**

**Perform Artificial Neural Network on the given datasets.**

**Use TensorFlow keras to build your model in Python and use Neural net package in R**

**Briefly explain the output in the documentation for each step in your own words.**

**Use different activation functions to get the best model.**

Now we split our data in X\_train, X\_test, Y\_train, Y\_test 85% data on train and 15

test . Preparing a ANN model on training data set And for tuning again split data set into 85% and 15% for validation data ,then test on test data.

Build and Train Model details Name -model

Multilayer perceptron model, with one hidden layer.

input layer : 4 neuron, represents the feature of Iris

hidden layer : 16 neuron, activation using ReLU

output layer : 3 neuron, represents the class of Iris, Softmax Layer

optimizer = stochastic gradient descent with no batch-size

loss function = categorical cross entropy

learning rate = default from keras.optimizer.SGD, 0.01

epoch = 200

train accuracy ~ 86

test accuracy for fresh data ~96

due to less data

# Classification Report:-

precision recall f1-score support

0 1.00 1.00 1.00 7

1 1.00 0.70 0.82 10

2 0.67 1.00 0.80 6

accuracy 0.87 23

macro avg 0.89 0.90 0.87 23

weighted avg 0.91 0.87 0.87 23

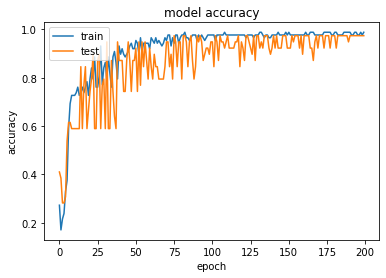
# Confusion Matrix

[[7 0 0]

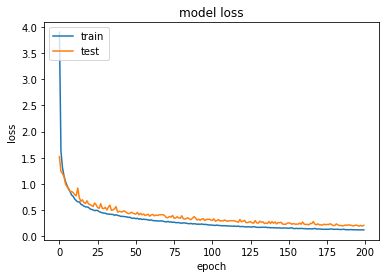
[0 7 3]

[0 0 6]]

#train vs. test accuracy plot with each epoch:-

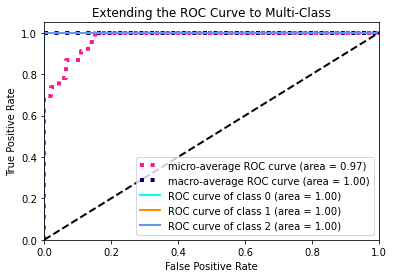


#training vs test loss plot:-



##Loss function plot:-

%



**And After this we build another model for selection of best Activation Function with different para meter.**

**Cross Validation Process model name - mlpc\_tuned**

**Parameters for CV created in dictionary structure**

**INFORMATION ABOUT THE INPUTED PARAMETERS**

**alpha: float, default = 0.0001 L2 penalty (regularization term) parameter. (penalty parameter)**

# Classification Report

precision recall f1-score support

0 1.00 1.00 1.00 7

1 1.00 0.80 0.89 10

2 0.75 1.00 0.86 6

accuracy 0.91 23

macro avg 0.92 0.93 0.92 23

weighted avg 0.93 0.91 0.91 23

**# Confusion Matrix**

**[[7 0 0]**

**[0 8 2]**

**[0 0 6]]**

For Forward propagate input to a network output & Backpropagate error and store in neurons and others I have written some custom function. Due to lack of time not able to implement .

Used library –

pandas for data manipulations

Numpy for Numerical Calculatations

Sklearn for Data mining / Machine learning

Matplotlib for Data visualization

Seaborn for Advance data visualization

Scipy for Advance data visualization

keras and tensorflow

itertools