# TUGAS 1.3: PROBABILISTIC NEURAL NETWORKS


Name = Puruso Muhammad Hanunggul

Class = IF-39-INT

NIM = 1301153680


## Problem Description (Case Study)

In this case, we were given 150 data train that has 4 variables, which are 3 data inputs and 1 data output. The output data is a class for each 3 dimensional data (data input) that consist of '0', '1', and '2'. Given 30 data test that has 3 data inputs. The machine should be able to classify (determine) which class for each data by using Probabilistic Neural Networks (PNN).

## Method and Design

This problem is solved by using PNN methods of learning there is a few step that in this method of learning to solve this problem. First step, the data that already we get should be specified. The input data should be in input layer. So, by using K-Fold method, I already create a data test and data train to predict the value.

Second step, after we get a data set and data train from data test, we have to find $\sigma$ (sigma) value. In this case, I use the first way to smoothing $\sigma$ value which is probability density function. To find best $\sigma$ value, we must find the accuracy of based on $\sigma$. By using brute force, I start from $\sigma = 0$ until $\sigma = 30$ which each iterative is increment by 0.2. To the accuracy we check each $\sigma$ into this formula:

$$g_i(\mathbf{x}) = e^{-\frac{\left|\left|\mathbf{x}-\mathbf{x}_i\right|\right|^2}{2\sigma^2}}$$

After we get g(x), find the **f(x) (summation)** which is sum all data which has a same class. In this case, we have three class, which are '0', '1', and '2'. Then, go back to the looping of the first step, generate new data training and data testing again. Since I use K-Fold K =3, therefore, every data test consist of 50 data and the rest will be data training. Hence, each iterative will has 3 accuracy. To compute accuracy is by comparing the correct data that has been get from **f(x)**, between predicted data and real data, then the amount of correctness is divided by whole data test.

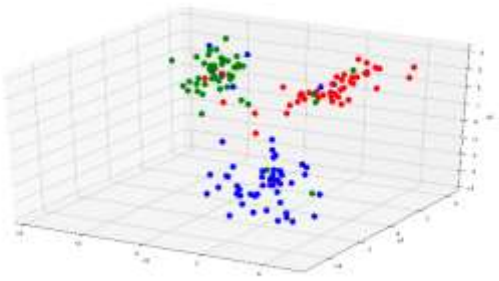
After we get 3 different accuracies, average it then we will get the accuracy for its iterative. Loop this algorithm until we get the upper limit of $\sigma$ which is 30, then we will get a lot of accuracies based on each iterative of $\sigma$.

Choose the highest accuracy, then use its $\sigma$ for classifying data set by using the same method (PNN)

Here is some screenshot and Figure of the program.

# Screenshot

*Data Scatter Plot*



**Red = Class 0**

**Green = Class 1**

**Blue = Class 2**

*Screenshot of the Algorithm*

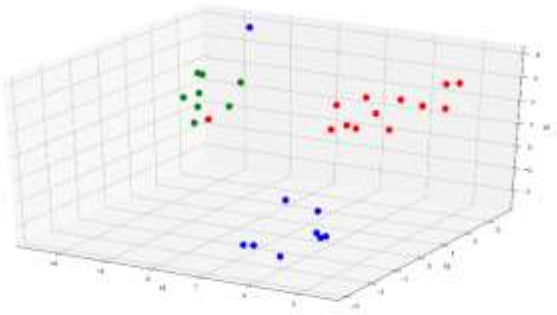
```
##############MAIN#########################
sett = train(0,0)

x1,x2,x3 =[],[],[]
for x in (sett):
    Xi = x[0]
    x1.append(Xi)
    Yi = x[1]
    x2.append(Yi)
    Zi = x[2]
    x3.append(Zi)
    a = x[3]
    label.append(a)
    if a == 0:
        ax.scatter(Xi, Yi, Zi, s=100, c='red')
    elif a == 1:
        ax.scatter(Xi, Yi, Zi, s=100, c='green')
    else:
        ax.scatter(Xi, Yi, Zi, s=100, c='blue')
    ax.set_xlabel('x1')
    ax.set_ylabel('x2')
    ax.set_zlabel('x3')
#####
calcTrainMain(0,50)
hasil = max(temp)
sigma = hasil[1]
print('hasil akurasi',hasil)
print('sigmanya',sigma)
print(len(x1))
m = main(sigma)
#####
for x in m:
    Xi = x[0]
    x1.append(Xi)
    Yi = x[1]
    x2.append(Yi)
    Zi = x[2]
    x3.append(Zi)
    ase = x[3]
    label.append(ase)
    if ase == 0.0:
        ax2.scatter(Xi, Yi, Zi, s=100, c='red')
    elif ase == 1.0:
        ax2.scatter(Xi, Yi, Zi, s=100, c='green')
    else:
        ax2.scatter(Xi, Yi, Zi, s=100, c='blue')
    ax2.set_xlabel('x1')
    ax2.set_ylabel('x2')
    ax2.set_zlabel('x3')
plt.show()
```

Here is some of the code. This screenshot shows the main Program of my code which has several function inside.

- calcTrainMain(x,y): is the function to compute the pdf, and also summation that incrementally increase 0.2 from the lower values, parameter x, to the upper value, parameter y.
- train(x,y): is a function to create a set of data from '.xlsx' file. The parameter x is the start column and the parameter y is used to determine the end column by using (nrows-y)
- main(sigma): after we got the best accuracy, we also get its sigma. The $\sigma$ is used for this function to classify each class in data set.
  - Inside the function of calcTrainMain(x,y), there is a function acc(ts) which compute the accuracy of data test (ts), and there is a function calcGx(a,b,c,s,sk,dSet) which is to compute the pdf of g(x) based on amount of x1,x2, x3 and using the brute force of $\sigma$. Parameter sk is a data train that is specify what is the length of its data, and dSet is used for smoothing. The rest, it can be checked inside the attached program.

*The Result of Data Test*



**Red = Class 0**

**Green = Class 1**

**Blue = Class 2**

The result of data test class can be checked inside **'Final Result Tugas 1.3 MacLearn Fixed.xlsx'**